

---

# TANDEM: Bi-Level Data Mixture Optimization with Twin Networks

---

Anonymous Author(s)

Affiliation

Address

email

## Abstract

1       The capabilities of large language models (LLMs) significantly depend on training  
2       data drawn from various domains. Optimizing domain-specific mixture ratios can  
3       be modeled as a bi-level optimization problem, which we simplify into a single-  
4       level penalized form and solve with twin networks: a proxy model trained on  
5       primary data and a dynamically updated reference model trained with additional  
6       data. Our proposed method, Twin Networks for bi-level Data mixture optiMiza-  
7       tion (TANDEM), measures the data efficacy through the difference between the  
8       twin models and up-weights domains that benefit more from the additional data.  
9       TANDEM provides theoretical guarantees and wider applicability, compared to  
10      prior approaches. Furthermore, our bi-level perspective suggests new settings to  
11      study domain reweighting such as data-restricted scenarios and supervised fine-  
12      tuning, where optimized mixture ratios significantly improve the performance.  
13      Extensive experiments validate TANDEM’s effectiveness in all scenarios.

## 14   1 Introduction

15   The success of large language models (LLMs) largely relies on extensive training data collected from  
16   diverse domains, including chat logs [16], academic writings [31], mathematical problems [43], and  
17   code repositories [19]. The emergent capabilities observed in LLMs are substantially influenced by  
18   the specific composition of cross-domain corpora [12, 37]. Therefore, it is important to carefully  
19   balance the proportions of domain-specific data in training sets to ensure models develop intended  
20   and balanced capabilities for target domains.

21   Optimizing the mixture ratios of data domains can be formulated as a bi-level optimization problem,  
22   in which the inner loop optimizes model parameters for a fixed ratio on training data, while the outer  
23   loop searches for the best mixture ratio on validation data. Due to the difficulty of exactly solving this  
24   bi-level problem, we transform it into a single-level optimization problem. Specifically, the inner-level  
25   optimization is viewed as a Lagrangian penalty within the outer-level objective. This perspective  
26   naturally motivates the introduction of twin networks: a proxy model trained exclusively on the  
27   primary training data, and a reference model exposed to additional validation data. Interestingly,  
28   this twin-network formulation relates closely to prior methods such as DoReMi [36] and DoGE  
29   [7], offering insights into addressing their limitations. Based on this formulation, we propose Twin  
30   Networks for bi-level Data mixture optiMization (TANDEM), an algorithm that measures the  
31   domain data efficacy through the twin models’ disparities and ultimately approximates the original  
32   bi-level optimization problem. Unlike DoReMi, which employs a static reference model, TANDEM  
33   dynamically updates both models. Additionally, TANDEM enhances stability and provides theoretical  
34   convergence guarantees compared to DoGE, as it aggregates multiple updating steps and avoids  
35   relying directly on gradient estimation, thereby mitigating issues related to high variance.

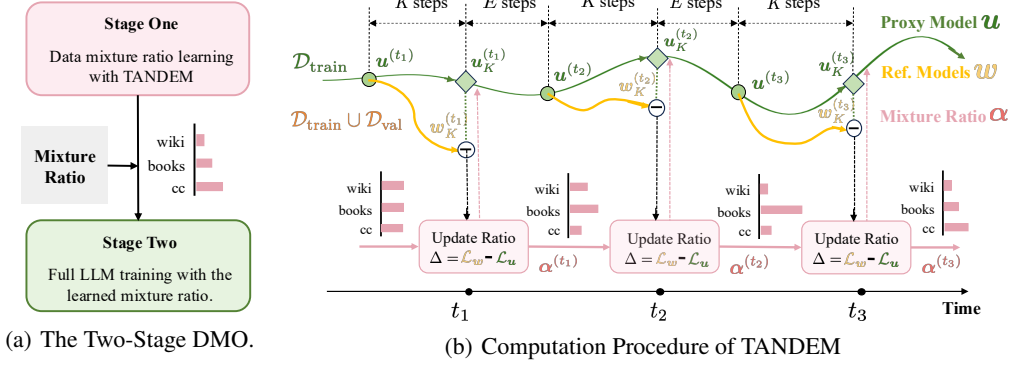


Figure 1: (a) The two-stage data mixture optimization. Optimal mixtures are first learned and then utilized to train the final model. (b) The computation procedure of TANDEM, twined proxy model (green) and reference model (orange) are used to determine the update of the mixture ratio (pink).

The bi-level formulation also emphasizes that future research on data domain weighting could focus more on scenarios with limited domain-specific data and supervised fine-tuning (SFT), rather than exclusively on traditional pretraining settings with abundant data. From a bi-level optimization viewpoint, assigning equal mixture ratios becomes a valid solution for single-epoch training when domain data is abundant, aligning with recent empirical findings highlighting uniform mixing strategies as competitive baselines [2]. Despite the prevalence of big data, limited data scenarios are quite common, particularly within specific domains, since large datasets frequently consist of many heterogeneous smaller datasets [32]. Furthermore, SFT often requires domain data to be visited multiple times, which creates generalization gap that leads to non-trivial solution for the bi-level problem. It is precisely in these cases that optimizing data mixture ratios can yield significant improvements.

While previous methods like DoReMi and those built on data mixing laws [39, 22, 14] are not directly applicable to these newer scenarios, TANDEM can be effectively extended. Our experimental results demonstrate TANDEM’s effectiveness in such settings.

Our contributions can be summarized as follows:

- We introduce TANDEM, an effective and efficient algorithm for data mixture optimization that utilizes twined proxy and reference networks to approximate the bi-level objective. TANDEM enjoys theoretical convergence guarantees.
- We highlight that data mixture optimization is particularly beneficial in scenarios with limited data availability rather than traditional pretraining setups with abundant domain data.
- We empirically demonstrate TANDEM’s effectiveness across standard and data-limited scenarios, showing its superiority over a set of competitive data mixture optimization methods.

## 2 Methodology

We formulate the problem of finding the optimal data mixture ratio as bi-level optimization. To solve the problem, we propose our penalty-based algorithm TANDEM, as presented in Figure 1(b) and Algorithm 1 in Appendix A. TANDEM draws insights from many previous works, while improving upon them. Besides, by inspecting the bilevel optimization formulation, we suggest broadening the scope of research on data mixture optimization under limited-domain data and supervised fine-tuning (SFT), rather than focusing solely on conventional data-rich pretraining settings. The notations used in this paper are summarized in Appendix F.

### 2.1 Problem Formulation

Consider training an LLM on a data composition from  $M$  domains,  $\{\mathcal{D}_1, \mathcal{D}_2, \dots, \mathcal{D}_M\}$  (e.g. Wikipedia, CommonCrawl). Data mixture optimization (DMO) refers to the problem of finding

the optimal proportions of data for each domain  $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_M]$  over probability simplex  $\mathcal{A} := \{\alpha \in \mathbb{R}^M \mid \sum_{m=1}^M \alpha_m = 1, \alpha_m \geq 0\}$ . For any data mixture ratio  $\alpha$ , and its corresponding model parameter  $w(\alpha)$  obtained by training loss, our goal is to minimize the validation loss  $\mathcal{L}_{\text{val}}(w(\alpha))$  over  $\alpha$ . The optimization problem is formulated as follows:

$$\min_{\alpha \in \mathcal{A}} \mathcal{L}_{\text{val}}(w(\alpha)) := \sum_{m=1}^M \mathcal{L}_{\text{val}}^m(w(\alpha)) \quad \text{s.t. } w(\alpha) \in \arg \min_w \mathcal{L}_{\text{train}}(\alpha, w) := \sum_{m=1}^M \alpha_m \mathcal{L}_{\text{train}}^m(w). \quad (1)$$

By splitting the data into training and validation sets, we can construct the validation and training loss  $\mathcal{L}_{\text{val}}^m(w)$  and  $\mathcal{L}_{\text{train}}^m(w)$  on domain  $\mathcal{D}_m$ . Intuitively, in the bi-level problem, the outer level problem seeks the optimal domain weights for validation loss with the model weights obtained by the inner level reweighted training loss. Similar to [36, 7], given the learned final mixture ratio  $\alpha^*$ , we construct the final training set by sampling  $\mathcal{D}_{\alpha^*} \triangleq \sum_{m=1}^M \alpha^* \cdot \text{UNIF}(\mathcal{D}_m)$  upon which the final model is trained (Figure 1(a)).

## 2.2 Twin Networks for Bi-level Data Mixture Optimization

Solving the bi-level optimization problem (1) is challenging due to its nested structure. By viewing the inner-level problem as a constraint and subsequently incorporating it into the outer-level problem as a Lagrangian penalty, (1) can be reformulated as a single-level problem [28, 17]:

$$\min_{\alpha \in \mathcal{A}, w} \mathcal{H}_\gamma(\alpha, w) := \mathcal{L}_{\text{val}}(w) + \gamma \left( \mathcal{L}_{\text{train}}(\alpha, w) - \min_u \mathcal{L}_{\text{train}}(\alpha, u) \right). \quad (2)$$

Here, the auxiliary variable  $u$  is introduced as a proxy of  $w(\alpha) \in S^*(\alpha) := \arg \min_w \mathcal{L}_{\text{train}}(\alpha, w)$ . The constrain in (1) is transferred into the penalization  $\mathcal{L}_{\text{train}}(\alpha, w) - \min_u \mathcal{L}_{\text{train}}(\alpha, u)$ . Clearly, by properly invoking  $\gamma \rightarrow \infty$ , the solution of (2) will approximate the original (1). This claim is justified by Proposition 3 in [28]. We refer readers to the Appendix A for more information.

Next, we proceed to illustrate our algorithm of optimizing penalized Lagrange problem (2).

**Algorithm Procedure** Besides the mixture ratio  $\alpha$ , optimizing (2) deals with two LM models: the proxy model  $u$  and a reference model  $w$ . As shown in Figure 1(b), the optimization processes of  $\alpha$ ,  $w$ , and  $u$  are indexed by  $t$ ,  $k$  and  $k$ .

**Update on  $u$ :** Firstly, given a data mixture ratio  $\alpha^{(t)}$ , we find the  $u \in \arg \min_u \mathcal{L}_{\text{train}}(\alpha^{(t)}, u)$  in the penalization term under certain mixture ratio  $\alpha$ . Our  $u$  is updated for  $K$  steps to approximate the optimal one:

$$u_{k+1}^{(t)} = u_k^{(t)} - \eta_u \nabla_u \mathcal{L}_{\text{train}}(\alpha^{(t)}, u_k^{(t)}) \quad (3)$$

(green line with arrow in Figure 1(b)). The proxy model is trained on the whole training set and maintained through the DMO process.

**Update on  $w$ :**  $w$  serves as a reference model updated on both the training and the validation sets. Similar to the proxy model  $u$ ,  $w$  is updated for  $K$  steps before one data mixture ratio  $\alpha$  update, which is used to optimize the inner problem of penalized problem (2):

$$w_{k+1}^{(t)} = w_k^{(t)} - \eta_w \left( \nabla_w \mathcal{L}_{\text{val}}(\alpha^{(t)}, w_k^{(t)}) + \gamma \nabla_w \mathcal{L}_{\text{train}}(\alpha^{(t)}, w_k^{(t)}) \right). \quad (4)$$

(orange line in Figure 1(b)). Intuitively, training  $w$  for multiple steps provides more subtle guidance for mixture ratio update as will be elaborated in the  $\alpha$  update part.

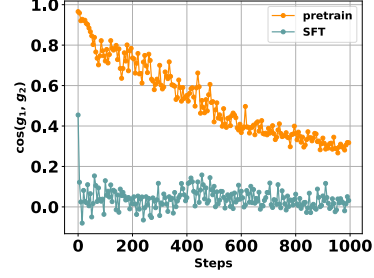
Unlike the proxy model  $u$ , which is maintained throughout training (green line in Figure 1(b)), we do not maintain an independent reference model  $w$ , but rather synchronize the starting point of  $w$  and  $u$  by setting  $w_0^{(t)} = u_0^{(t)}$  as a initialization of the  $K$   $w$  updates. By doing so, we control the disparity between  $w$  and  $u$  during the optimization. Intuitively,  $w$  and  $u$  should not diverge from each other as  $u$  acts as a proxy of  $w(\alpha) \in S^*(\alpha)$  and  $w$  approximates  $w_\gamma^*(\alpha) \in S_\gamma^*(\alpha) := \arg \min_w \mathcal{H}_\gamma(\alpha, w)$ . Clearly, the ideal  $w_\gamma^*(\alpha)$  under penalized problem will approximate the  $w^\alpha$  when  $\gamma \rightarrow \infty$ , since the penalized Lagrange problem (2) approximates the original problem (1).

**Update on  $\alpha$ :** The mixture ratio update in (5) seeks a solution of (2). That says, a data mixture ratio yields a trained model with good performance on data from the validation set under all domains. The

Table 1: Summary of  $\Delta$  in DoReMi, DoGE and TANDEM. All three methods update  $\alpha$  with two models.

Method	Hyper-gradient $\Delta$
DoReMi	$-\max\{\underbrace{\mathcal{L}_{\text{train}}^{1:M}(\alpha^{(t)}, \mathbf{u}^{(t)})}_{\text{proxy model}} - \underbrace{\mathcal{L}_{\text{train}}^{1:M}(\bar{\alpha}, \bar{\mathbf{w}})}_{\text{reference model}}, 0\}$
DoGE	$\underbrace{\mathcal{L}_{\text{train}}^{1:M}(\alpha^{(t)}, \mathbf{u}^{(t)})}_{\text{reference model}} - \eta \nabla \mathcal{L}_{\text{val}}(\mathbf{u}^{(t)}) - \underbrace{\mathcal{L}_{\text{train}}^{1:M}(\alpha^{(t)}, \mathbf{u}^{(t)})}_{\text{proxy model}}$
TANDEM	$\underbrace{\mathcal{L}_{\text{train}}^{1:M}(\alpha^{(t)}, \mathbf{w}_K^{(t)})}_{\text{reference model}} - \underbrace{\mathcal{L}_{\text{train}}^{1:M}(\alpha^{(t)}, \mathbf{u}_K^{(t)})}_{\text{proxy model}}$

Figure 2: SFT exhibits higher gradient variance than pretraining



111 update rule of is given by the following projected gradient descent:

$$\alpha^{(t+1)} = \Pi_{\mathcal{A}} \left( \alpha^{(t)} - \eta_{\alpha} \gamma \left( \underbrace{\mathcal{L}_{\text{train}}^{1:M}(\alpha^{(t)}, \mathbf{w}_K^{(t)})}_{\text{reference model}} - \underbrace{\mathcal{L}_{\text{train}}^{1:M}(\alpha^{(t)}, \mathbf{u}_K^{(t)})}_{\text{proxy model}} \right) \right) \quad (5)$$

112 (Pink line in Figure 1(b).) The post-updated  $\mathbf{u}_K^{(t)}$  and  $\mathbf{w}_K^{(t)}$  are applied to capture the domain-wise loss  
 113 difference  $\mathcal{L}_{\text{train}}^m(\mathbf{w}_K^{(t)}) - \mathcal{L}_{\text{train}}^m(\mathbf{u}_K^{(t)})$ . Since the  $\mathbf{u}_K^{(t)}$  and  $\mathbf{w}_K^{(t)}$  are respectively trained on training  
 114 set and training set plus validation set, the aforementioned gap captures the gain of incorporating the  
 115 additional validation data. Larger loss differences indicate that the model gets significant improved  
 116 by consuming more data, thus the corresponding domains are up-weighted.

117 In each episode  $t$ ,  $K$  steps training on  $\mathbf{u}$  and  $\mathbf{w}$  are conducted to probe the proper direction of  $\alpha$   
 118 update. Notably, the proxy model  $\mathbf{u}$  and reference model  $\mathbf{w}$  are synchronized at the beginning of  
 119 probing, forming a Twin Networks for bi-level Data mixture optiMization (TANDEM) framework.  
 120 Since the updating of  $\alpha$  requires  $\mathbf{u}, \mathbf{w}$  probing, in practice, we decrease the frequency of updating  $\alpha$   
 121 to reduce the computational cost, and leave the proxy model  $\mathbf{u}$  trained freely for  $E$  steps before the  
 122 next  $\alpha$  update. Altogether the updates of  $\mathbf{u}, \mathbf{w}, \alpha$ , the data mixture optimization (DMO) problem  
 123 can be solved efficiently. The overall computation graph of TANDEM is outlined in Figure 1(b), and  
 124 a detailed workflow is summarized in Algorithm 1 in Appendix A.

125 **Convergence Analysis** Next, we explore the convergence rate of our proposed method. For a  
 126 non-convex bi-level optimization problem, it is standard to study its first-order stationary convergence  
 127 result e.g., [28, 17]. Notably, our problem (2) is a constrain problem over  $\alpha \in \mathcal{A}$ . Thus, it should be  
 128 considered in first-order stationary condition as in [8, 28]. The convergence result of our method is  
 129 summarized in the following theorem.

130 **Theorem 1** (Informally). *For sufficiently large  $T$ , under mild assumptions (Detailed in Appendix A),  
 131 for  $\alpha^{(t)}$  obtained in TANDEM Algorithm 1, the problem (1) converge to its first-order stationary  
 132 point in the rate of  $\mathcal{O}(T^{-\frac{1}{4}})$  by properly selecting  $\gamma, K, E, \eta_{\alpha}, \eta_{\mathbf{w}},$  and  $\beta_{\mathbf{u}}$ .*

133 The proof is left in Appendix A. Theorem 1 indicates that TANDEM theoretically ensures the  
 134 optimality of the learned mixture ratio.

### 135 2.3 TANDEM Improves Existing DMO Methods

136 We discuss the relationship between TANDEM and two existing methods, DoReMi [36] and DoGE [7].  
 137 By comparing the hyper-gradient  $\Delta$  that determines  $\alpha$  updates in different methods, we show that  
 138 our TANDEM draws common insights from previous works and improves upon them.

139 DoReMi updates the mixture ratio according to the excess loss of a proxy model  $\mathbf{u}$  relative to a  
 140 reference model  $\bar{\mathbf{w}}$  trained on uniformly sampled data. DoGE [7] pioneers bilevel optimization to  
 141 settle data mixtures and tracks the influence [26] of each domain on the validation data. Specifically,

<sup>1</sup>DoReMi and DoGE utilize exponential gradient descent to update  $\alpha$  where  $\alpha^{(t+1)} \propto \alpha^{(t)} \exp(-\eta_{\alpha} \Delta)$

142  $\Delta_{\text{DoGE}} = \langle \nabla \mathcal{L}_{\text{train}}(\alpha^{(t)}, u^{(t)}), \nabla \mathcal{L}_{\text{val}}(\alpha^{(t)}, u^{(t)}) \rangle$ . The inner product  $\langle \cdot \rangle$  is essentially a first-  
 143 order approximation of the domain-wise loss difference between a proxy model and a reference  
 144 model, where the reference model is obtained by one-step update on the validation data.

145 Outlined in Table 1, we see that  $\Delta$  of all three methods takes the form of per-domain loss difference  
 146 between a proxy model and a reference model. Nevertheless, contrasting DoReMi which adopts  
 147 a fixed reference model, the reference model  $w$  in TANDEM is dynamic, which better captures  
 148 the current training status. In its original form, DoGE incurs significant computational and memory  
 149 overhead as it maintains the per-domain gradients. Relying explicitly on the gradient estimation  
 150 makes it vulnerable to instability issues in high gradient variance scenarios like supervised fine-  
 151 tuning (SFT). Our TANDEM, in the contrary, better adapts, as increasing the probing step  $K$  helps  
 152 reduce the variance on  $\Delta$  as will be elaborated in Section 4.3. In Figure 2, we show that SFT exhibits  
 153 higher gradient variance than pretraining. The alignment of  $g_1$  and  $g_2$   $\cos(g_1, g_2)$  is used as a proxy  
 154 of the gradient variance, where  $g_1$  and  $g_2$  are gradients evaluated on different batches and lower  
 155 alignment  $\cos(g_1, g_2)$  indicates larger gradient variance.

## 156 2.4 Domain Reweighting Beyond Traditional Settings

157 In the above, we discuss our method to design the data mixture ratio by solving bi-level problem (1).  
 158 The circumstances under which DMO is most effective remain to be explored. Theoretically, the  
 159 standard training setting sets  $\alpha_m = 1/M$  for every  $m$ . Let us check the following proposition.

160 **Proposition 1.** Assume  $\mathcal{L}_{\text{train}}^m = \mathcal{L}_{\text{val}}^m$ , the uniform data mixture ratio  $\bar{\alpha}_m = \frac{1}{M}$  for  $m = 1, \dots, M$   
 161 constitutes a valid solution of the bilevel mixture optimization (1).

162 The proof is left in Appendix B. As can be seen, when the generalization gap  $|\mathcal{L}_{\text{val}}^m - \mathcal{L}_{\text{train}}^m|$  goes to  
 163 zero, uniform weighting emerges as a valid solution, so that the reweighting becomes less significant.  
 164 This aligns with the empirical observations that uniform weighting is highly competitive that many  
 165 DMO methods can not consistently outperform [2]. In conventional pretraining scenarios with  
 166 abundant data in each domain, the generalization gap is indeed small under one-epoch training.

167 However, when there is limited data in specific domains or the trained model overfits to some domain,  
 168 since both phenomena result in a large generalization gap, the reweighting technique over data  
 169 domains becomes significant. In fact, even though LLMs are trained on massive datasets overall, it is  
 170 common for specific domains to be relatively small e.g., specialized scientific literature, low-resource  
 171 languages, or domain-specific user interactions. In practice, the data-restricted scenario is ubiquitous.  
 172 Furthermore, in SFT, repeated passes over the same domain data exacerbate overfitting and widen the  
 173 generalization gap [9], which presents more interesting opportunities for domain reweighting.

## 174 3 Related Work

175 Data mixture optimization is drawing increasing attention to design LLMs with comprehensive  
 176 and balanced capabilities. Our work follows the recent trend of formulating the DMO as a bilevel  
 177 optimization. We summarize the related literature from these two strands of research in the following.

178 **Data Mixture Optimization** Conventional industry practices determine the optimal cross-domain  
 179 composition with human expertise [6, 33, 10]. To circumvent the exhaustive trial-and-error, various  
 180 heuristics has been explored. DoReMi [36] settles the mixture ratio by minimizing the worst-domain  
 181 excess loss relative to a well-trained reference model, perusing good performance in all domains.  
 182 [22, 39, 14] fit global data mixing laws, predict the performance on other mixtures and search for  
 183 those with good performances. Skill-It [3] trains several models to fit an inter-domain relation matrix,  
 184 which is later used to establish a local data mixing law that predicts the per-domain loss given certain  
 185 data mixture and the current model loss. Aioli [2] improves Skill-It by dynamically updating the  
 186 inter-domain relation matrix according to current model states. Nevertheless, these approaches remain  
 187 theoretically ungrounded due to their inductive nature, and incorporating the fitting of the mixing  
 188 law incurs additional approximation error. DOGE [7] pioneers bi-level optimization to settle data  
 189 mixtures and tracks the influence [26] of each domain on the validation data. However, assessing the  
 190 influence relies directly on the per-domain gradient estimation, which undermines its efficacy in high  
 191 gradient variance scenarios while incurring large computational and memory overhead.

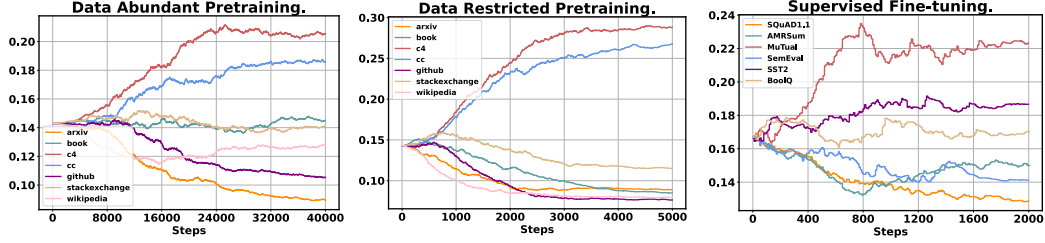


Figure 3: Step-wise data mixture ratio evolution under three scenarios. (a) data-abundant pretraining (b) data-restricted pretraining and (c) supervised fine-tuning.

**Bilevel Optimization** Bilevel optimization has been an important research topic in many scientific disciplines, however, solving the bi-level optimization problem is challenging due to the complicated dependency of the upper-level and lower-level problems. Typical bi-level optimization algorithms [23, 5, 27, 11, 4, 13] requires estimation of the implicit gradient, which requires second-order derivatives on the lower-level variables. Incorporating the Hessian makes it computationally prohibitive for large-scale problems. Recently [28, 17, 18] pioneer the penalized methods, where the inner-level problem is reformulated into an optimality constraint. As first-order gradient-based approaches, the penalized methods avoid the estimation of Hessian or Jacobian. Though effective in theory, their practical applications in large-scale LLM settings remain rarely explored. Our TANDEM belongs to this category, and is specially tailored for large-scale data mixture optimization problems.

## 4 Experiments

In this section, we first compare TANDEM to state-of-the-art algorithms in Section 4.2. Then we analyse the effectiveness of each design ingredient in Section 4.3. Code will be released upon acceptance.

### 4.1 Experimental Setup

We consider three application scenarios, conventional data-abundant pretraining, data-restricted training, and supervised fine-tuning. To the best of our knowledge, this is the first work to explore DMO in supervised fine-tuning. A brief summarization of the experimental setup is introduced below, while complete hyper-parameter settings and implementation details are in Appendix C.

**Data-Abundant Scenario:** For the data-abundant scenario, we train 160M GPT-style LMs [11] on a 6B sampled version of SlimPajama [29] as in [2]. SlimPajama consists of 7 domains: ArXiv, Books, CommonCrawl, C4, Github, StackExchange, and Wikipedia. The statistics of this sampled corpus are given in Figure 4. We set  $E = 20$ ,  $K = 5$ , train with batch size 8 and context length 2048 for 40000 steps (with respect to updates of proxy model  $u$ , so the mixture ratio  $\alpha$  is updated for  $\sim 2000$  steps.) as [2]. Though the SlimPajama-6B corpus exhibits significant domain imbalance, 40K steps of training doesn't deplete even the smallest domain, so this setting constitutes a data-abundant one-epoch scenario. The penalty constant  $\gamma$  is set to 1 across all the experiments. Experiments are conducted on eight NVIDIA Hopper H-100s.

**Data-Restricted Scenario:** For the data-restricted scenario, we construct a 300M sampled version of SlimPajama and keep the domain distribution unchanged. GPT-style LMs of size 160M, 410M and 1B are trained to examine the scalability of TANDEM. In this scenario, we train with  $K = E = 5$ , batch size 128, and context length 512 for 5000 steps, which ensures that samples in small domains like Arxiv, Books, StackExchange, and Wikipedia are exposed more than once. After DMO, the learned mixture ratio is utilized to resample the 300M corpus for the final model training.

**Supervised Fine-tuning:** For supervised fine-tuning, we select 6 tasks from Natural Instructions [24, 34], SQuAD1.1, AMRSum, MuTual, SemEval, SST2 and BoolQ. Prioritizing diversity of capabilities and formats, this corpus is comprised of open-ended text generation, multiple choice, and True/False tasks. Each task contains  $\sim 6000$  samples, where we split 1000 samples for test. In this



Table 2: Comparison for the 160M GPT-style model on SlimPajama in the data-abundant scenario (Upper) and data-restricted scenario (Lower). Per-domain perplexity is reported and “Average” represents the exponential of the average loss across all domains as in [7, 2]. † denotes the results reported use the mixture ratio given in Aioli [2].

	Methods	Arxiv	Book	C4	CommonCrawl	Github	Stackexchange	Wikipedia	Average
Data Abundant Regime	Uniform	<b>11.46</b>	62.53	66.43	59.29	6.71	13.98	28.31	25.74
	DoReMi†	12.71	80.09	82.60	71.76	5.75	14.26	29.54	28.32
	DoGE†	12.89	<b>51.50</b>	<b>54.32</b>	<b>49.34</b>	8.48	16.77	37.21	26.60
	Skill-It†	11.76	62.24	64.58	59.84	<b>6.36</b>	<b>12.36</b>	34.87	25.87
	Aioli†	11.47	61.89	65.52	58.24	6.74	14.08	28.48	25.66
	TANDEM	11.53	61.82	65.92	58.86	6.63	13.76	<b>27.26</b>	<b>25.43</b>
Data Restricted Regime	Uniform	18.05	65.86	71.05	63.76	9.37	17.94	34.27	31.53
	DoReMi	18.90	80.29	89.05	79.02	10.24	19.74	43.20	36.91
	DoGE	17.76	60.88	65.08	58.81	9.00	17.71	33.94	30.10
	Skill-It	20.93	<b>52.00</b>	57.11	<b>49.50</b>	<b>8.77</b>	<b>16.74</b>	40.49	29.24
	Aioli	17.68	62.48	69.02	61.44	9.26	17.79	33.06	30.67
	TANDEM	<b>16.85</b>	52.75	<b>56.82</b>	51.11	8.99	18.21	<b>32.52</b>	<b>28.07</b>

scenario, we train a pretrained Qwen2-500M model [38] with  $K = 20$ ,  $E = 10$ , batch size 32, and context length 512 for 2000 steps. Different from pretraining where the majority of samples are only trained once, in instruction tuning, each sample is on average exposed 2.5 times.

## 4.2 Comparisons with State of the Arts

We compare TANDEM against various state-of-the-art methods. **Uniform** A simple baseline that uniformly mixes groups and requires zero extra training runs. **DoReMi** [36] adopts the idea of distributionally robust optimization and searches for the data mixture ratio by minimizing the worst-domain excess loss over a reference model trained with the uniform strategy. **DOGE** [7] solves the DMO problem by tracking the data influence of each domain on the validation set and up-weights the most influential domains. **Skill-It** [3] trains several models to fit an inter-domain relation matrix, which is later used to establish an incremental data mixing law that induces a mixture ratio  $\alpha$  update rule. **Aioli** [2] improves Skill-It by dynamically updating the inter-domain relation matrix according to current model states. For the baselines, We use the published implementations.<sup>2</sup> Averaged results from 3 runs are reported. Due to limited space, the standard deviation is given in Appendix D.

**Data-Abundant Pretraining** In Table 2 (Upper), we see that TANDEM discovers mixture ratios with comparative performance. As discussed in Section 2.4, in this scenario, the uniform strategy is highly competitive. The mixture ratio for the baselines is obtained from Aioli [2]. We show the step-wise data mixture ratio evolution during TANDEM in Figure 3 (Left). Note that the uniform strategy is not the only valid solution to the DMO problem, and TANDEM finds another solution that performs equally well. The detailed learned mixture ratio of each method is given in Appendix E.

**Data-Restricted Pretraining** For the data-restricted training scenario, TANDEM significantly outperforms baselines as shown in Table 2 (Lower). For instance, it achieves 28.07 averaged perplexity, surpassing the most competitive Skill-It by 1.17 and the uniform baseline by 3.46. In this scenario, the uniform strategy is no longer competitive. Equally assigning  $\frac{1}{M}$  weights potentially leads to overfitting in small domains (repeated multiple times), while leaving the large domains underfitting. The mixture ratios learned with TANDEM and other baselines are shown in Figure 5, and the step-wise data mixture ratio evolution during the data mixture optimization is given in Figure 3 (Middle). We see that after DMO, CommonCrawl and C4 take the majority, driven by their extensive lexical diversity and complex semantics. Nevertheless, compared to the original data distribution, TANDEM up-weights the small domains Arxiv, Books, Github, StackExchange and Wikipedia from 3.4%, 3.7%, 4.2%, 2.8%, 3.1% to 8.9%, 8.5%, 7.6%, 11.5%, 7.9% respectively, preventing these small domains from being overwhelmed while avoiding potential overfitting. Besides, we inspect the scalability

<sup>2</sup>**DoReMi**: <https://github.com/sangmichaelxie/doremi>, **DOGE**: <https://github.com/Olivia-fsm/DoGE>, **Skill-It** and **Aioli**: <https://github.com/HazyResearch/aioli>

Figure 4: SlimPajama-6B Statistics.

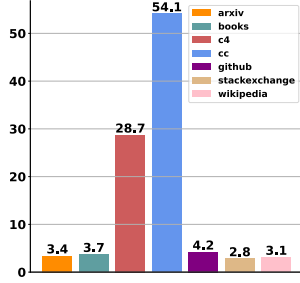


Table 3: Comparison for models of different sizes.

	160M		410M		1B	
	Avg.	Time	Avg.	Time	Avg.	Time
Uniform	31.53	-	29.59	-	29.91	-
DoReMi	36.91	16 <sub>min</sub>	54.61	31 <sub>min</sub>	56.53	41 <sub>min</sub>
DoGE	30.10	65 <sub>min</sub>	27.45	172 <sub>min</sub>	-	-
Skill-It	29.24	28 <sub>min</sub>	27.70	59 <sub>min</sub>	27.15	80 <sub>min</sub>
Aioli	31.19	36 <sub>min</sub>	28.79	64 <sub>min</sub>	28.07	93 <sub>min</sub>
TANDEM	<b>28.07</b>	26 <sub>min</sub>	<b>25.00</b>	57 <sub>min</sub>	<b>24.35</b>	77 <sub>min</sub>

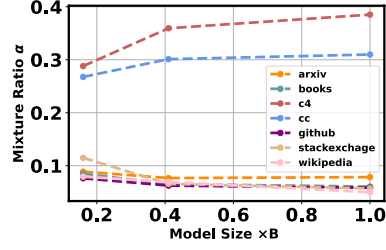
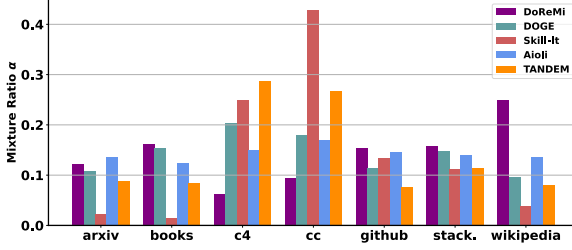


Figure 5: Mixture ratio learned by different methods. Figure 6: Impact of different model sizes.

of TANDEM with three different scales (160M, 410M, 1B) in Table 3. TANDEM consistently outperforms the baselines with large margin while not incurs much computational overhead. We omit DoGE for the 1B model experiment due to its large memory consumption.

**Supervised Fine-tuning** The results for supervised fine-tuning are given in Table 4. For the text generation tasks SQuAD1.1 and AMRSum, we report the Rouge-L [20] score. For the multi-choice tasks (MuTual, SemEval) and Yes/No tasks (SST2, BoolQ), the Accuracy is reported. We also outline the averaged metric value as well as the test loss. TANDEM outperforms all the baselines. In SFT, the improvement in test loss is more significant than that in the final evaluation metrics. We hypothesize this is caused by the imperfect alignment between them [21, 30].

### 4.3 Analyses

To evaluate the effectiveness of each design component, we conduct ablation experiments with the 160M GPT-style model.

**The Effect of Synchronizing  $u$  and  $w$**  One obvious characteristic of our method is that the proxy model  $u$  and the reference model  $w$  are synchronized by setting  $w_0^{(t)} = u_0^{(t)}$ . We show the effect of the synchronization by inspecting  $\text{Dist}(u, w) = \|u - w\|_2$  along the DMO training process. From Figure 7(a), we see that with synchronization, the distance between  $u$  and  $w$  is well controlled under  $1.5e^{-4}$  and gradually contracts during the whole DMO process. This contraction is critical for  $w$  and  $\alpha$  in the penalized problem (2) converges to the original bi-level optimization problem (1).

Table 4: Comparison for the 500M Qwen-2 model on a mixture of 6 supervised fine-tuning tasks.

Method	SQuAD1.1	AMRSum	MuTual	SemEval	SST2	BoolQ	Avg. Metric $\uparrow$	Test Loss $\downarrow$
Uniform	72.62	45.18	72.72	<b>89.75</b>	87.75	80.29	74.72	0.591
DoReMi	71.40	43.40	70.97	88.50	87.00	77.43	73.11	0.686
DoGE	71.26	44.81	71.67	89.00	88.30	<b>81.04</b>	74.35	0.563
Skill-It	72.14	44.21	73.07	89.40	88.40	80.34	74.60	0.539
Aioli	72.35	45.01	73.57	89.10	88.10	79.64	74.63	0.542
TANDEM	<b>72.73</b>	<b>45.19</b>	<b>73.77</b>	89.70	<b>88.70</b>	80.04	<b>75.03</b>	<b>0.508</b>



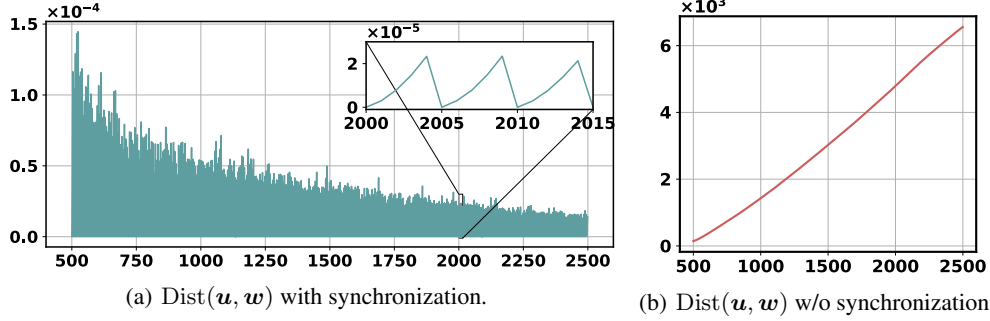


Figure 7: The  $\text{Dist}(\mathbf{u}, \mathbf{w})$  evolution comparison during DMO with and without  $\mathbf{u}, \mathbf{w}$  synchronization.

as discussed in Section 2. On the contrary, independently maintaining the proxy model  $\mathbf{u}$  and the reference model  $\mathbf{w}$  incurs blown-up distance  $\text{Dist}(\mathbf{u}, \mathbf{w})$  as shown in Figure 7(b).

**The Effect of the Probing Steps  $K$**  During DMO, the hyper-gradient  $\Delta$  determines the update of  $\alpha$ . To validate the effectiveness of  $K$  in reducing the variance of  $\Delta$ , we trace  $\cos(\Delta, \tilde{\Delta})$  through the training.  $\tilde{\Delta}$  is the hyper-gradient evaluated using another batch of data other than that of  $\Delta$ , so  $\cos(\Delta, \tilde{\Delta})$  serves as a proxy of the variance, the better  $\Delta$  and  $\tilde{\Delta}$  aligns, the smaller the variance of hyper-gradient is. We inspect under the SFT setting where the model parameters exhibited large variance. During the training,  $\alpha$  is fixed to prevent the interference of inaccurate mixture ratio.

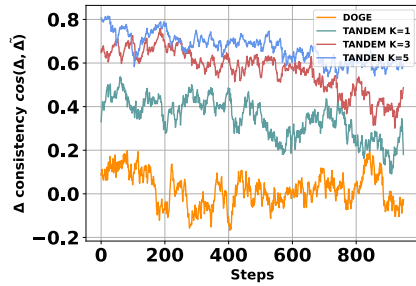


Figure 8: The variance of  $\Delta$  decreases with more probing steps.

From Figure 8, we see that DoGE exhibits the largest  $\Delta$  variance as it explicitly depends on the noisy parameter gradient estimation. As  $K$  increases, the variance of  $\Delta$  decrease, leading to more reliable update of the mixture ratio. Nevertheless, large  $K$  increases the computational cost. So in practice and we need to deliberately choose the proper  $K$ .

**The Effect of the Model Scale** To investigate how the model size will impact the final mixture ratio. In Figure 6, we compare  $\alpha$  learned with models of size 160M, 410M, and 1B. We see that learned  $\alpha$  with larger models are slightly "sharper" than the smaller ones. More specifically, the 1B model further increases the weights of the already large CommonCrawl and C4 while down-weights the others. For large models, due to the increasing capability of memorizing samples, smaller domains are less likely to be overwhelmed, while the risk of potential overfitting increases. The capability of capturing this subtle difference further demonstrates the effectiveness of TANDEM. Nevertheless, the optimal mixture ratios under different model scales share the same trend, so a smaller model can work as a valid proxy for efficient searching.

## 5 Conclusion

In this paper, we propose TANDEM, a principled, efficient and versatile data mixture optimization framework. By solving the DMO bilevel optimization problem, TANDEM ensures the optimality of the learned mixture ratios, along with a  $\mathcal{O}(T^{-\frac{1}{4}})$  convergence rate. Besides the algorithmic contribution, from the bilevel optimization perspective, we further demonstrate the limitation of the conventional data-abundant setting in DMO and advocate new settings like data-restricted scenario as well as supervised fine-tuning. Extensive experiments and analysis are conducted to demonstrate the effectiveness of our approach. Our work deepens the understanding of data mixture optimization and expands its application scenarios.

## References

- [1] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O’Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, and Edward Raff et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, 2023.
- [2] Mayee F. Chen, Michael Y. Hu, Nicholas Lourie, Kyunghyun Cho, and Christopher Ré. Aioli: A unified optimization framework for language model data mixing. In *International Conference on Learning Representations*, 2025.
- [3] Mayee F. Chen, Nicholas Roberts, Kush Bhatia, Jue Wang, Ce Zhang, Frederic Sala, and Christopher Ré. Skill-it! a data-driven skills framework for understanding and training language models. In *Neural Information Processing Systems*, 2023.
- [4] Tianyi Chen, Yuejiao Sun, and Wotao Yin. Closing the gap: Tighter analysis of alternating stochastic gradient methods for bilevel problems. In *Advances in Neural Information Processing Systems*, 2021.
- [5] Sang Keun Choe, Sanket Vaibhav Mehta, Willie Neiswanger Hwijeen Ahn, Pengtao Xie, Emma Strubell, and Eric Xing. Making scalable meta learning practical. In *Advances in Neural Information Processing Systems*, 2024.
- [6] Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P Bosma, Zongwei Zhou, Tao Wang, Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. GLaM: Efficient scaling of language models with mixture-of-experts. In *Proceedings of the 39th International Conference on Machine Learning*, 2022.
- [7] Simin Fan, Matteo Pagliardini, and Martin Jaggi. Doge : Domain reweighting with generalization estimation. In *International Conference on Machine Learning*, 2024.
- [8] Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1):267–305, 2016.
- [9] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. Book in preparation for MIT Press.
- [10] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, and Abhishek Kadian et al. The llama 3 herd of models. *arXiv preprint*, arXiv:2407.21783, 2024.
- [11] Riccardo Grazzi, Massimiliano Pontil, and Saverio Salzo. Bilevel optimization with a lower-level contraction: Optimal sample complexity without warm-start. In *Journal of Machine Learning Research*, 2023.
- [12] Shangmin Guo, Yi Ren, Stefano V Albrecht, and Kenny Smith. Sample relationships through the lens of learning dynamics with label information. In *Workshop on Interpolation Regularizers and Beyond at NeurIPS*, 2022.
- [13] Mingyi Hong, Hoi-To Wai, Zhaoran Wang, and Zhuoran Yang. A two-timescale stochastic algorithm framework for bilevel optimization: Complexity analysis and application to actor-critic. *SIAM J. Optim.*, 2023.
- [14] Feiyang Kang, Yifan Sun, Bingbing Wen, Si Chen, Dawn Song, Rafid Mahmood, and Ruoxi Jia. Autoscale: Automatic prediction of compute-optimal data compositions for training LLMs. *arXiv preprint*, arXiv:2407.20177, 2025.
- [15] Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-Łojasiewicz condition. In *European Conference on Machine Learning*, 2016.

- [16] Andreas Köpf, Yannic Kilcher, Dimitri von Rütte, Sotiris Anagnostidis, Zhi Rui Tam, Keith Stevens, Abdullah Barhoum, Duc Minh Nguyen, Oliver Stanley, Richárd Nagyfi, Shahul ES, Sameer Suri, David Alexandrovich Glushkov, Arnav Varma Dantuluri, Andrew Maguire, Christoph Schuhmann, Huu Nguyen, and Alexander Julian Mattick. Openassistant conversations - democratizing large language model alignment. In *Neural Information Processing Systems*, 2023.
- [17] Jeongyeol Kwon, Dohyun Kwon, Stephen Wright, and Robert Nowa. A fully first-order method for stochastic bilevel optimization. In *International Conference on Machine Learning*, 2023.
- [18] Jeongyeol Kwon, Dohyun Kwon, Stephen Wright, and Robert D Nowak. On penalty methods for nonconvex bilevel optimization and first-order stochastic approximation. In *International Conference on Learning Representations*, 2024.
- [19] Raymond Li, Loubna Ben allal, and Yangtian Zi et.al. Starcoder: may the source be with you! *Transactions on Machine Learning Research*, 2023.
- [20] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, 2004.
- [21] Hong Liu, Sang Michael Xie, Zhiyuan Li, and Tengyu Ma. Same pre-training loss, better downstream: Implicit bias matters for language models. In *International Conference on Machine Learning*, 2023.
- [22] Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. Regmix: Data mixture as regression for language model pre-training. In *International Conference on Learning Representations*, 2025.
- [23] Dagr  ou Mathieu, Pierre Ablin, Samuel Vaiter, and Thomas Moreau. A framework for bilevel optimization that enables stochastic and global variance reduction algorithms. In *Advances in Neural Information Processing Systems*, 2022.
- [24] Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. Cross-task generalization via natural language crowdsourcing instructions. In *Association for Computational Linguistics*, 2022.
- [25] Maher Nouiehed, Maziar Sanjabi, Tianjian Huang, Jason D Lee, and Meisam Razaviyayn. Solving a class of non-convex min-max games using iterative first order methods. *Advances in Neural Information Processing Systems*, 2019.
- [26] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems*, 2020.
- [27] Amirreza Shaban, Ching-An Cheng, Nathan Hatch, and Byron Boots. Truncated back-propagation for bilevel optimization. In *International Conference on Artificial Intelligence and Statistics*, 2019.
- [28] Han Shen and Tianyi Chen. On penalty-based bilevel gradient descent method. In *International Conference on Machine Learning*, 2023.
- [29] Daria Soboleva, Faisal Al-Khateeb, Joel Hestness, and Jacob Robert Steeves Nolan Dey Open-tensor: Robert Myers. Slimpajama: A 627b token, cleaned and deduplicated version of redpajama. <https://www.cerebras.ai/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpajama>, 2023.
- [30] Yi Tay, Mostafa Dehghani, Samira Abnar, Hyung Won Chung, William Fedus, Jinfeng Rao, Sharan Narang, Vinh Q. Tran, Dani Yogatama, and Donald Metzler. Scaling laws vs model architectures: How does inductive bias influence scaling? In *Conference on Empirical Methods in Natural Language Processing*, 2023.
- [31] Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. Galactica: A large language model for science. *arXiv preprint*, arXiv:2211.09085, 2022.

- 411 [32] Yee Whye Teh. On big data learning for small data problems. KDD '18, page 3, New York, NY,  
412 USA, 2018. Association for Computing Machinery.
- 413 [33] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timo-  
414 thée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez,  
415 Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation  
416 language models. *arXiv preprint*, arXiv:2302.13971, 2023.
- 417 [34] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei,  
418 Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, and David Stap  
419 et al. Super-natural instructions: generalization via declarative instructions on 1600+ tasks. In  
420 *Conference on Empirical Methods in Natural Language Processing.*, 2022.
- 421 [35] Quan Xiao, Songtao Lu, and Tianyi Chen. A generalized alternating method for bilevel learning  
422 under the polyak-Łojasiewicz condition. In *Neural Information Processing Systems*, 2023.
- 423 [36] Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang,  
424 Quoc V Le, Tengyu Ma, and Adams Wei Yu. Doremi: Optimizing data mixtures speeds up  
425 language model pretraining. In *Neural Information Processing Systems*, 2023.
- 426 [37] Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy Liang. Data selection for language  
427 models via importance resampling. In *Neural Information Processing Systems*, 2023.
- 428 [38] An Yang, Baosong Yang, and Binyuan Hui et.al. Qwen2 technical report. *arXiv preprint*,  
429 arXiv:2407.10671, 2024.
- 430 [39] Jiasheng Ye, Peiju Liu, Tianxiang Sun, Jun Zhan, Yunhua Zhou, and Xipeng Qiu. Data mixing  
431 laws: Optimizing data mixtures by predicting language modeling performance. In *International  
432 Conference on Learning Representations*, 2025.
- 433 [40] Mingyang Yi, Lu Hou, Jiacheng Sun, Lifeng Shang, Xin Jiang, and Qun Liu. Improved ood  
434 generalization via adversarial training and pretraing. In *International Conference on Machine  
435 Learning*, 2021.
- 436 [41] Mingyang Yi, Ruoyu Wang, and Zhi-Ming Ma. Characterization of excess risk for locally  
437 strongly convex population risk. *Advances in Neural Information Processing Systems*, 2022.
- 438 [42] Mingyang Yi, Ruoyu Wang, Jiacheng Sun, Zhenguo Li, and Zhi-Ming Ma. Breaking cor-  
439 relation shift via conditional invariant regularizer. In *International Conference on Learning  
440 Representations*, 2023.
- 441 [43] Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok,  
442 Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical  
443 questions for large language models. In *International Conference on Learning Representations*,  
444 2024.

## A Convergence of Bi-level Data Mixture Optimization

### A.1 The Proposed Algorithm

As mentioned in the main part of this paper, the data mixture optimization problem is bi-level. Here we present a theoretical analysis of the proposed TANDEM. The original bi-level optimization problem

$$\mathcal{P} : \min_{\alpha \in \mathcal{A}} \mathcal{L}_{\text{val}}(\mathbf{w}^*(\alpha)) \quad \text{s.t. } \mathbf{w}^*(\alpha) \in S^*(\alpha) := \arg \min_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) \quad (6)$$

is difficult to solve owing to the imposed hard constraint. We turn to the Lagrangian problem of  $\mathcal{P}$  such that

$$\mathcal{P}_\gamma : \min_{\alpha \in \mathcal{A}, \mathbf{w}} \mathcal{L}_{\text{val}}(\mathbf{w}) + \gamma \left( \mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) - \min_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) \right) \quad (7)$$

There is a vast variety of existing literature e.g., [28, 17, 35] discuss the relationship between the Lagrangian problem  $\mathcal{P}_\gamma$  and the original problem  $\mathcal{P}$ . In a word, when taking  $\gamma$  sufficient large, the solution of  $\mathcal{P}_\gamma$  will approximate the solution of  $\mathcal{P}$ . Thereafter, we can develop the algorithm to  $\mathcal{P}_\gamma$  to solve  $\mathcal{P}$ , as we did in this paper.

---

#### Algorithm 1 Twin Networks for bi-level Data mixture optimization (TANDEM)

---

**Input:**

Train set  $\mathcal{D}_{\text{train}}$ , validation set  $\mathcal{D}_{\text{val}}$  comprised of  $M$  domains.  
 Episode number  $T$ , Episode length  $E$ , Probing length for each episode  $K$ ,  
 Learning rate  $\eta_w, \eta_u, \eta_\alpha$  for  $\mathbf{w}$  (reference),  $\mathbf{u}$  (proxy) and  $\alpha$  (mixture) respectively.  
 1: Initialize proxy model parameters  $\mathbf{u}^0$  and domain weights  $\alpha^0$ .  
 2: **for**  $t = 0$  **to**  $T - 1$  **do**

*// Mixture ratio  $\alpha$  update.*

3: Set  $\mathbf{w}_0^{(t)}, \mathbf{u}_0^{(t)} \leftarrow \mathbf{u}^{(t)}$ .  
 4: **for**  $k = 0$  **to**  $K - 1$  **do**  
 5:      $\mathbf{u}_{k+1}^{(t)} = \mathbf{u}_k^{(t)} - \eta_u \nabla_{\mathbf{u}} \mathcal{L}_{\text{train}}(\alpha^{(t)}, \mathbf{u}_k^{(t)})$ .  
 6:      $\mathbf{w}_{k+1}^{(t)} = \mathbf{w}_k^{(t)} - \eta_w (\nabla_{\mathbf{w}} \mathcal{L}_{\text{val}}(\mathbf{w}_k^{(t)}) + \gamma \nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha^{(t)}, \mathbf{w}_k^{(t)}))$ .  
 7: **end for**  
 8:      $\alpha^{(t+1)} = \Pi_{\mathcal{A}}(\alpha^{(t)} - \eta_\alpha (\underbrace{\mathcal{L}_{\text{train}}^{1:M}(\alpha^{(t)}, \mathbf{w}_K^{(t)})}_{\text{reference model}} - \underbrace{\mathcal{L}_{\text{train}}^{1:M}(\alpha^{(t)}, \mathbf{u}_K^{(t)})}_{\text{proxy model}}))$

*// Free model weights  $\mathbf{u}$  update.*

9: **for**  $e = 0$  **to**  $E - 1$  **do**  
 10:      $\mathbf{u}_{K+e+1}^{(t)} = \mathbf{u}_{K+e}^{(t)} - \eta_u \nabla_{\mathbf{u}} \mathcal{L}_{\text{train}}(\alpha^{(t+1)}, \mathbf{u}_{K+e}^{(t)})$ .  
 11: **end for**  
 12: Set  $\mathbf{u}^{(t+1)} \leftarrow \mathbf{u}_{K+E}^{(t)}$ .  
 13: **end for**

**Output:**

model weights  $\mathbf{w}_K^{(T)}$ , domain weights  $\alpha^{(T)}$ .

---

A full workflow of Twin Networks for bi-level Data mixture optimization (TANDEM) is shown in Algorithm 1. TANDEM alternates between updating the mixture ratio  $\alpha$  and the proxy model  $\mathbf{u}$ , whereas  $\mathbf{u}$  is used to approximate the minimum of  $\mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$ . Update of the mixture ratio  $\alpha$  requires probing the data efficacy of each domain. During probing, we optimize the reference model  $\mathbf{w}$ , as well as the proxy model  $\mathbf{u}$  for  $K$  steps.  $\mathbf{w}$  and  $\mathbf{u}$  are respectively trained on the train set and the train set plus validation set, their incurred loss gap captures the gain of incorporating the additional validation data. TANDEM then up-weights domains that benefit more from additional data. Notably, the proxy model  $\mathbf{u}$  and reference model  $\mathbf{w}$  are synchronized at the beginning of probing. Since the updating of  $\alpha$  requires  $\mathbf{u}, \mathbf{w}$  probing, we decrease the frequency of updating  $\alpha$  to reduce the computational cost, and leave the  $\mathbf{u}$  trained freely for  $E$  steps before the next  $\alpha$  update.

### A.2 Convergence Rate

Next, we explore the convergence rate of the proposed Algorithm 1. For the original problem  $\mathcal{P}$  (6) its convergence property under the iterates obtained by solving the Lagrange problem (7) has been

explored, under different regularity conditions [28, 17]. In this paper, we will present our result under a mild Polyak-Łojasiewicz (PL) condition [41, 15, 28, 42, 25], which has been widely imposed to study the bi-level optimization problem. Technically, we will impose the following Assumptions to derive the convergence rate. Before illustrating our Assumptions, we need the following definitions to simplify the notations. We denote

$$\mathcal{H}_\gamma(\alpha, \mathbf{w}) = \mathcal{L}_{\text{val}}(\mathbf{w}) + \gamma \left( \mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) - \min_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) \right), \quad (8)$$

with  $S_\gamma^*(\alpha) = \{\mathbf{w} : \mathbf{w} \in \arg \min_{\mathbf{w}} \mathcal{H}_\gamma(\alpha, \mathbf{w})\}$ , and define  $S^*(\alpha) = \{\mathbf{w} : \mathbf{w} \in \arg \min_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w})\}$  and

$$\mathcal{H}(\alpha) = \inf_{\mathbf{w} \in S^*(\alpha)} \mathcal{L}_{\text{val}}(\mathbf{w}), \quad (9)$$

where  $\mathcal{H}(\alpha)$  is well-defined since  $\mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$  is continuous to  $\mathbf{w}$ . Besides that, the minimum of  $\mathcal{H}(\alpha)$  is exactly the solution to original problem  $\mathcal{P}$  [6].

**Assumption 1** (PL condition). *For any  $\alpha \in \mathcal{A}$ , both  $\mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$  and  $\mathcal{H}_\gamma(\alpha, \mathbf{w})$  satisfy the PL inequality with coefficient  $\mu$  and  $\mu_\gamma$ , respectively. That says:*

$$\mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) - \min_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) \leq \frac{1}{2\mu} \|\nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w})\|^2 \quad (10)$$

and

$$\mathcal{H}_\gamma(\alpha, \mathbf{w}) - \min_{\mathbf{w}} \mathcal{H}_\gamma(\alpha, \mathbf{w}) \leq \frac{1}{2\mu_\gamma} \|\nabla \mathcal{H}_\gamma(\alpha, \mathbf{w})\|^2. \quad (11)$$

Moreover, the coefficient  $\mu_\gamma$  satisfies  $\lim_{\gamma \rightarrow \infty} \frac{\mu_\gamma}{\gamma} = 1$ .

**Assumption 2** (Smoothness). *For any  $\alpha \in \mathcal{A}$ ,*

1. *Both  $\nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$  and  $\nabla_{\mathbf{w}} \mathcal{L}_{\text{val}}(\mathbf{w})$  are Lipschitz continuous to  $\alpha$  (hold for  $\nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$ ) and  $\mathbf{w}$  on coefficient  $L$ .*
2. *For any  $\alpha \in \mathcal{A}$ , both  $\mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$  and  $\mathcal{L}_{\text{val}}(\mathbf{w})$  are Lipschitz continuous to  $\mathbf{w}$  with coefficient  $B$ .*

**Assumption 3** (Bounded Hessian). *For any  $\alpha \in \mathcal{A}$ ,  $\mathbf{w} \in S^*(\alpha)$ , and positive constants  $\lambda, \rho$ , the Hessian matrices  $\nabla_{\mathbf{w}\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) \succeq \lambda \mathbf{I}$ <sup>3</sup> and  $\nabla_{\alpha\mathbf{w}}^2 \mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) \preceq \rho$ .*

**Assumption 4** (Lipschitz Hessian). *For any  $\alpha \in \mathcal{A}$ ,  $\mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$  is two-times continuous differentiable, and the Hessian matrices  $\nabla_{\alpha\mathbf{w}}^2 \mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$ ,  $\nabla_{\mathbf{w}\mathbf{w}}^2 \mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$  are all Lipschitz continuous to  $\mathbf{w}$  with coefficient  $H$ .*

**Assumption 5** (Bounded Loss Function). *The non-negative loss function  $\mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$ ,  $\mathcal{L}_{\text{val}}(\mathbf{w})$  are uniformly bounded by positive constant  $D$ .*

Notably, for the bounded Hessian condition, due to the structure of  $\mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$ , it can be implied by the lower bounded  $\nabla_{\mathbf{w}\mathbf{w}} \mathcal{L}_{\text{train}}^m(\mathbf{w})$  and an upper bounded  $\nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\mathbf{w})$  for any  $m$ . Moreover, it worth noting that for bi-level optimization problem, it is standard to impose some regularity conditions to the Hessian matrix. For examples, the smooth Hessian in [28, 17] and lower bounded Hessian in [17]. Therefore, the imposed bounded Hessian Assumption 3 can be considered as a mild condition.

Next, we present a lemma to characterize the gap between the gradient of  $\nabla_{\alpha} \mathcal{H}(\alpha)$  and  $\nabla_{\alpha} \mathcal{H}_\gamma(\alpha, \mathbf{w})$ .

**Lemma 1.** *Under Assumptions 1-4 for any given  $\alpha$  and  $\mathbf{w}_\gamma^*(\alpha) \in S_\gamma^*(\alpha)$ , it holds*

$$\left\| \nabla_{\alpha} \mathcal{H}(\alpha) - \frac{\partial}{\partial \alpha} \mathcal{H}_\gamma(\alpha, \mathbf{w}_\gamma^*(\alpha)) \right\| \leq \frac{1}{\gamma} \left( \frac{HB^2}{\mu^2} \left( \frac{\rho}{\lambda} + 1 \right) + \frac{\rho LB}{\mu\lambda} \right) \quad (12)$$

*Proof.* Without loss of generality, suppose that  $\mathcal{H}(\alpha) = \mathcal{L}_{\text{val}}(\mathbf{w}^*(\alpha))$ , due to the chain rule, we have

$$\nabla_{\alpha} \mathcal{H}(\alpha) = \nabla_{\mathbf{w}} \mathcal{L}_{\text{val}}(\mathbf{w}^*(\alpha)) = \nabla_{\alpha} \mathbf{w}^*(\alpha)^\top \nabla_{\mathbf{w}} \mathcal{L}_{\text{val}}(\mathbf{w}^*(\alpha)). \quad (13)$$

<sup>3</sup>For two matrices  $\mathbf{A}$ ,  $\mathbf{A} \succeq \lambda \mathbf{I}$  means  $\mathbf{A} - \lambda \mathbf{I}$  is a positively semi-definite matrix.



503 To simplify the notation, we denote  $w_\gamma^*(\alpha)$  and  $w^*(\alpha)$  as  $w_\gamma^*$  and  $w^*$  in the sequel. On the penalized  
504 problem, for any  $w_\gamma^*(\alpha)$ , we have

$$\begin{aligned}
& \frac{\partial}{\partial \alpha} \mathcal{H}_\gamma(\alpha, w_\gamma^*) \\
&= \nabla_\alpha \mathcal{H}_\gamma(\alpha, w_\gamma^*) + \nabla_\alpha w_\gamma^{*\top} \nabla_w \mathcal{H}_\gamma(\alpha, w_\gamma^*) \\
&= \nabla_\alpha \mathcal{H}_\gamma(\alpha, w_\gamma^*) \\
&= \gamma (\nabla_\alpha \mathcal{L}_{\text{train}}(\alpha, w_\gamma^*) - \nabla_\alpha \mathcal{L}_{\text{train}}(\alpha, w^*) - \nabla_\alpha w^{*\top} \nabla_w \mathcal{L}_{\text{train}}(\alpha, w^*)) \\
&= \gamma (\nabla_\alpha \mathcal{L}_{\text{train}}(\alpha, w_\gamma^*) - \nabla_\alpha \mathcal{L}_{\text{train}}(\alpha, w^*) - \nabla_{\alpha w}^2 \mathcal{L}_{\text{train}}(\alpha, w^*)(w_\gamma^* - w^*)) \\
&\quad - \nabla_\alpha w^{*\top} (\nabla_w \mathcal{L}_{\text{val}}(w^*) + \gamma \nabla_w \mathcal{L}_{\text{train}}(\alpha, w^*)) \\
&\quad + \nabla_\alpha w^{*\top} \nabla_w \mathcal{L}_{\text{val}}(w^*) + \gamma \nabla_{\alpha w}^2 \mathcal{L}_{\text{train}}(\alpha, w^*)(w_\gamma^* - w^*).
\end{aligned} \tag{14}$$

505 Besides that, we have

$$\begin{aligned}
& \nabla_\alpha w^{*\top} (\nabla_w \mathcal{L}_{\text{val}}(w^*) + \gamma \nabla_w \mathcal{L}_{\text{train}}(\alpha, w^*)) \\
&= \nabla_\alpha w^{*\top} (\nabla_w \mathcal{L}_{\text{val}}(w^*) - \nabla_w \mathcal{L}_{\text{val}}(w_\gamma^*)) \\
&\quad + \gamma \nabla_\alpha w^{*\top} (\nabla_w \mathcal{L}_{\text{train}}(\alpha, w^*) - \nabla_w \mathcal{L}_{\text{train}}(\alpha, w_\gamma^*) + \nabla_{ww}^2 \mathcal{L}_{\text{train}}(\alpha, w^*)(w_\gamma^* - w^*)) \\
&\quad + \gamma \nabla_{\alpha w}^2 \mathcal{L}_{\text{train}}(\alpha, w^*)(w_\gamma^* - w^*),
\end{aligned} \tag{15}$$

506 due to the optimal conditions  $\nabla_w \mathcal{L}_{\text{val}}(w_\gamma^*) + \gamma \nabla_w \mathcal{L}_{\text{train}}(\alpha, w_\gamma^*) = 0$ , and  $\nabla_{\alpha w}^2 \mathcal{L}_{\text{train}}(\alpha, w^*) +$   
507  $\nabla_\alpha w^{*\top} \nabla_{ww}^2 \mathcal{L}_{\text{train}}(\alpha, w^*) = 0$ . Combining the two above equations and (13), we get

$$\begin{aligned}
& \left\| \frac{\partial}{\partial \alpha} \mathcal{H}_\gamma(\alpha, w_\gamma^*) - \nabla_\alpha \mathcal{H}(\alpha) \right\| \\
&\leq \left\| \gamma (\nabla_\alpha \mathcal{L}_{\text{train}}(\alpha, w_\gamma^*) - \nabla_\alpha \mathcal{L}_{\text{train}}(\alpha, w^*) - \nabla_{\alpha w}^2 \mathcal{L}_{\text{train}}(\alpha, w^*)(w_\gamma^* - w^*)) \right\| \\
&\quad + \left\| \gamma \nabla_\alpha w^{*\top} (\nabla_w \mathcal{L}_{\text{train}}(\alpha, w^*) - \nabla_w \mathcal{L}_{\text{train}}(\alpha, w_\gamma^*) + \nabla_{ww}^2 \mathcal{L}_{\text{train}}(\alpha, w^*)(w_\gamma^* - w^*)) \right\| \\
&\quad + \left\| \nabla_\alpha w^{*\top} (\nabla_w \mathcal{L}_{\text{val}}(w^*) - \nabla_w \mathcal{L}_{\text{val}}(w_\gamma^*)) \right\| \\
&\leq \gamma H \left( \frac{\rho}{\lambda} + 1 \right) \|w^* - w_\gamma^*\|^2 + \frac{\rho L}{\lambda} \|w^* - w_\gamma^*\|,
\end{aligned} \tag{16}$$

508 due to the bounded Hessian Assumption [3], Smoothness Assumption [2], and Lipchitz Assumption  
509 [4]. Then, due to the PL condition [1] and Smoothness Assumption [2], we know there exists a  $w_\gamma^*$  (the  
510 projection of  $w^*$  to  $S_\gamma^*(\alpha)$ ) satisfies

$$\begin{aligned}
\|w^* - w_\gamma^*\| &\leq \frac{1}{\mu} \|\nabla_w \mathcal{L}_{\text{train}}(\alpha, w_\gamma^*)\| \\
&\leq \frac{1}{\gamma \mu} (\|\nabla_w \mathcal{L}_{\text{val}}(w_\gamma^*) + \gamma \nabla_w \mathcal{L}_{\text{train}}(\alpha, w_\gamma^*)\| + \|\nabla_w \mathcal{L}_{\text{val}}(w_\gamma^*)\|) \\
&= \frac{\|\nabla_w \mathcal{L}_{\text{val}}(w_\gamma^*)\|}{\gamma \mu} \\
&\leq \frac{B}{\gamma \mu}.
\end{aligned} \tag{17}$$

511 Combining this with inequality (16), we obtain the conclusion under such  $w_\gamma^*$ . Finally, due to  
512 the Lemma A.5 in [25], we know that  $\mathcal{H}_\gamma(\alpha, w_\gamma^*)$  is invariant over  $w_\gamma^* \in S_\gamma^*(\alpha)$ , we prove our  
513 conclusion.  $\square$

514 From Lemma [1], we know that the gap between the gradients of original problem  $\mathcal{H}(\alpha)$  and its  
515 Lagrange version  $\mathcal{H}_\gamma(\alpha, w_\gamma^*(\alpha))$  can be extremely small by invoking penalty parameter  $\gamma \rightarrow \infty$ .  
516 Thus, it implies that we can compute the gradient of Lagrange problem to implement the gradient-  
517 based method. Next, we illustrate an useful lemma, which characterizes the (semi)-Lipschitz  
518 continuity of  $w_\gamma^*(\alpha)$  and  $w^*(\alpha)$  to  $\alpha$ .

519 **Lemma 2.** Under Assumptions [1](#) and [2](#) for any  $\alpha_1, \alpha_2 \in \mathcal{A}$ , it holds

$$\|\mathbf{w}^*(\alpha_1) - \mathbf{w}^*(\alpha_2)\| \leq \frac{L}{\mu} \|\alpha_1 - \alpha_2\|, \quad (18)$$

520 for any  $\mathbf{w}^*(\alpha_1) \in S^*(\alpha)$  and  $\mathbf{w}^*(\alpha_2) \in S^*(\alpha_2)$  satisfies  $\mathbf{w}^*(\alpha_2) = \arg \min_{\mathbf{w} \in S^*(\alpha_2)} \|\mathbf{w} -$   
 521  $\mathbf{w}^*(\alpha_1)\|$ . On the other hand, it holds

$$\|\mathbf{w}_\gamma^*(\alpha_1) - \mathbf{w}_\gamma^*(\alpha_2)\| \leq \frac{\gamma L}{\mu_\gamma} \|\alpha_1 - \alpha_2\| \quad (19)$$

522 for any  $\mathbf{w}_\gamma^*(\alpha_1) \in S_\gamma^*(\alpha_1)$  and some  $\mathbf{w}_\gamma^*(\alpha_2) = \arg \min_{\mathbf{w} \in S_\gamma^*(\alpha_1)} \|\mathbf{w} - \mathbf{w}_\gamma^*(\alpha_1)\|$ .

523 *Proof.* The two conclusions are directly obtained from Lemma A.3 in [\[25\]](#), we prove the second  
 524 conclusion due to the formulation of  $\nabla_{\mathbf{w}} \mathcal{H}_\gamma(\alpha, \mathbf{w})$ , we know

$$\begin{aligned} \|\nabla_{\mathbf{w}} \mathcal{H}_\gamma(\alpha_1, \mathbf{w}_\gamma^*(\alpha_2))\| &= \|\nabla_{\mathbf{w}} \mathcal{H}_\gamma(\alpha_1, \mathbf{w}_\gamma^*(\alpha_2)) - \nabla_{\mathbf{w}} \mathcal{H}_\gamma(\alpha_2, \mathbf{w}_\gamma^*(\alpha_2))\| \\ &= \gamma \|\nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha_1, \mathbf{w}_\gamma^*(\alpha_2)) - \nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha_2, \mathbf{w}_\gamma^*(\alpha_2))\| \\ &\leq \gamma L \|\alpha_1 - \alpha_2\|, \end{aligned} \quad (20)$$

525 due to Assumption [2](#) On the other hand, by invoking the Assumption [1](#) and [\(17\)](#)

$$\begin{aligned} \|\nabla_{\mathbf{w}} \mathcal{H}_\gamma(\alpha_1, \mathbf{w}_\gamma^*(\alpha_2))\| &= \|\nabla_{\mathbf{w}} \mathcal{L}_{\text{val}}(\mathbf{w}_\gamma^*(\alpha_2)) + \gamma \nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha_1, \mathbf{w}_\gamma^*(\alpha_2))\| \\ &\geq \mu_\gamma \|\mathbf{w}_\gamma^*(\alpha_2) - \mathbf{w}_\gamma^*(\alpha_1)\|. \end{aligned} \quad (21)$$

526 where  $\mathbf{w}_\gamma^*(\alpha_1)$  is the projection of  $\mathbf{w}^*(\alpha_2)$  to  $S_\gamma^*(\alpha_1)$ . By combining the two above inequalities, we  
 527 obtain our second conclusion. The first conclusion can be similarly proved.  $\square$

528 From this lemma, we can obtain the following Lipschitz smoothness of  $\mathcal{H}_\gamma(\alpha, \mathbf{w}_\gamma^*(\alpha))$  by the  
 529 following Lemma. It worth noting that  $\mathcal{H}_\gamma(\alpha, \mathbf{w}_\gamma^*(\alpha))$  is invariant over  $\mathbf{w}_\gamma^*(\alpha) \in S_\gamma^*(\alpha)$  as discussed  
 530 in the proof of Lemma [1](#). Thus, the  $\nabla_\alpha \mathcal{H}_\gamma(\alpha, \mathbf{w}_\gamma^*(\alpha))$  is well-defined.

531 **Lemma 3.** Under Assumptions [1](#) and [2](#)  $\nabla_\alpha \mathcal{H}_\gamma(\alpha, \mathbf{w}_\gamma^*(\alpha))$  has semi-Lipschitz gradient such that

$$\|\nabla_\alpha \mathcal{H}_\gamma(\alpha_1, \mathbf{w}_\gamma^*(\alpha_1)) - \nabla_\alpha \mathcal{H}_\gamma(\alpha_2, \mathbf{w}_\gamma^*(\alpha_2))\| \leq \gamma B \underbrace{\left( \frac{\gamma L}{\mu_\gamma} + \frac{L}{\mu} \right)}_{L_\gamma} \|\alpha_1 - \alpha_2\|. \quad (22)$$

532 *Proof.* From [\(14\)](#), we see

$$\begin{aligned} &\|\nabla_\alpha \mathcal{H}_\gamma(\alpha_1, \mathbf{w}_\gamma^*(\alpha_1)) - \nabla_\alpha \mathcal{H}_\gamma(\alpha_2, \mathbf{w}_\gamma^*(\alpha_2))\| \\ &\leq \gamma \|\nabla_\alpha \mathcal{L}_{\text{train}}(\alpha_1, \mathbf{w}_\gamma^*(\alpha_1)) - \nabla_\alpha \mathcal{L}_{\text{train}}(\alpha_2, \mathbf{w}_\gamma^*(\alpha_2))\| \\ &\quad + \gamma \|\nabla_\alpha \mathcal{L}_{\text{train}}(\alpha_2, \mathbf{w}_\gamma^*(\alpha_2)) - \nabla_\alpha \mathcal{L}_{\text{train}}(\alpha_1, \mathbf{w}_\gamma^*(\alpha_1))\| \\ &\leq \gamma B (\|\mathbf{w}_\gamma^*(\alpha_1) - \mathbf{w}_\gamma^*(\alpha_2)\| + \|\mathbf{w}_\gamma^*(\alpha_2) - \mathbf{w}_\gamma^*(\alpha_1)\|) \\ &\leq \gamma B \left( \frac{\gamma L}{\mu_\gamma} + \frac{L}{\mu} \right) \|\alpha_1 - \alpha_2\|, \end{aligned} \quad (23)$$

533 which proves our conclusion.  $\square$

534 Notably, for the original optimization problem, there exists a constrain  $\alpha \in \mathcal{A}$ . Thus, to prove  
 535 the first order stationary condition of constrain problem  $\min_{\alpha \in \mathcal{A}} \mathcal{H}_\alpha$ , we consider the generalized  
 536 projected gradient stable condition, i.e.,  $\frac{1}{\eta_\alpha} \|\alpha^{(t)} - \Pi_{\mathcal{A}}(\alpha^{(t)} - \eta_\alpha \nabla_\alpha \mathcal{H}(\alpha^{(t)}))\| \leq \epsilon$  for some small  
 537 negative  $\epsilon$ . This is a standard first order stationary condition for Non-convex optimization problem  
 538 with constrains [\[28, 40, 8\]](#). Due to Lemma [1](#) we know that

$$\begin{aligned} &\frac{1}{\eta_\alpha} \left\| \alpha^{(t)} - \Pi_{\mathcal{A}}(\alpha^{(t)} - \eta_\alpha \nabla_\alpha \mathcal{H}(\alpha^{(t)})) \right\| - \frac{1}{\eta_\alpha} \left\| \alpha^{(t)} - \Pi_{\mathcal{A}}(\alpha^{(t)} - \eta_\alpha \nabla_\alpha \mathcal{H}_\gamma(\alpha^{(t)}, \mathbf{w}_\gamma^*(\alpha^{(t)}))) \right\| \\ &\leq \frac{1}{\eta_\alpha} \left\| \Pi_{\mathcal{A}}(\alpha^{(t)} - \eta_\alpha \nabla_\alpha \mathcal{H}(\alpha^{(t)})) - \Pi_{\mathcal{A}}(\alpha^{(t)} - \eta_\alpha \nabla_\alpha \mathcal{H}_\gamma(\alpha^{(t)}, \mathbf{w}_\gamma^*(\alpha^{(t)}))) \right\| \\ &\leq \left\| \nabla_\alpha \mathcal{H}(\alpha^{(t)}) - \nabla_\alpha \mathcal{H}_\gamma(\alpha^{(t)}, \mathbf{w}_\gamma^*(\alpha^{(t)})) \right\| \\ &\leq \mathcal{O} \left( \frac{1}{\gamma} \right), \end{aligned} \quad (24)$$

when  $\gamma \rightarrow \infty$ , where the second inequality is from the Lipschitz continuity of projection operation [8]. Thus, the above inequality indicates that to prove the first order stationary condition of  $\mathcal{H}(\alpha)$ , it is sufficient to prove the first order stationary condition of  $\mathcal{H}_\gamma(\alpha, w)$ . Next we present our main theorem, i.e., the formal version of Theorem 1 which shows the convergence result of  $\mathcal{H}_\alpha(\alpha)$  under Algorithm 1.

**Theorem 1.** For sufficiently large  $T$ , under Assumptions 1-5 for the  $\alpha^{(t)}$  obtained in Algorithm 1 it holds

$$\min_{1 \leq t \leq T} \frac{1}{\eta_\alpha} \|\alpha^{(t)} - \Pi_{\mathcal{A}}(\alpha^{(t)} - \eta_\alpha \nabla_\alpha \mathcal{H}(\alpha^{(t)}))\| \leq \mathcal{O}\left(T^{-\frac{1}{4}}\right) \quad (25)$$

by selecting  $\gamma = \sqrt{T}$ ,  $K \geq \frac{\log \frac{\mu T^{-\frac{3}{4}}}{2D(1+\gamma)}}{\log(1-\frac{\mu}{L\gamma})}$ ,  $E \geq \min\left\{1, \frac{\log \frac{\mu \gamma T^{-\frac{3}{4}}}{2D}}{\log(1-\frac{\mu}{L})} - K\right\}$ ,  $\eta_\alpha = \frac{1}{4L_\gamma}$ ,  $\eta_w = \frac{1}{L_\gamma}$ ,  $\eta_u = \frac{1}{L}$ .

*Proof.* As mentioned in [24], it is sufficient to prove the first-order stationary condition for  $\mathcal{H}_\gamma(\alpha, w)$ . Due to the Lipschitz smoothness of it, we have

$$\begin{aligned} & \mathcal{H}_\gamma(\alpha^{(t+1)}, w_\gamma^*(\alpha^{(t+1)})) - \mathcal{H}_\gamma(\alpha^{(t)}, w_\gamma^*(\alpha^{(t)})) \\ & \leq \left\langle \nabla_\alpha \mathcal{H}_\gamma(\alpha^{(t)}, w_\gamma^*(\alpha^{(t)})), \alpha^{(t+1)} - \alpha^{(t)} \right\rangle + \frac{L_\gamma}{2} \|\alpha^{(t+1)} - \alpha^{(t)}\|^2 \\ & = \left\langle \nabla_\alpha \mathcal{F}_\gamma(\alpha^{(t)}, w_K^{(t)}, u_K^{(t)}), \alpha^{(t+1)} - \alpha^{(t)} \right\rangle \\ & + \left\langle \nabla_\alpha \mathcal{H}_\gamma(\alpha^{(t)}, w_\gamma^*(\alpha^{(t)})) - \nabla_\alpha \mathcal{F}_\gamma(\alpha^{(t)}, w_K^{(t)}, u_K^{(t)}), \alpha^{(t+1)} - \alpha^{(t)} \right\rangle \\ & + \frac{L_\gamma}{2} \|\alpha^{(t+1)} - \alpha^{(t)}\|^2 \\ & \leq \left( \frac{L_\gamma}{2} - \frac{1}{2\eta_\alpha} \right) \|\alpha^{(t+1)} - \alpha^{(t)}\|^2 + \frac{\eta_\alpha}{2} \left\| \nabla_\alpha \mathcal{H}_\gamma(\alpha^{(t)}, w_\gamma^*(\alpha^{(t)})) - \nabla_\alpha \mathcal{F}_\gamma(\alpha^{(t)}, w_K^{(t)}, u_K^{(t)}) \right\|^2, \end{aligned} \quad (26)$$

where the last inequality is due to the property of projection operator and Jensen's inequality. Let us define  $\bar{\alpha}^{(t+1)} = \Pi_{\mathcal{A}}(\alpha^{(t)} - \eta_\alpha \nabla_\alpha \mathcal{H}_\gamma(\alpha^{(t)}, w_\gamma^*(\alpha^{(t)})))$ , we proceed to upper bound the gap between  $\|\bar{\alpha}^{(t+1)} - \alpha^{(t)}\|$  and  $\|\alpha^{(t+1)} - \alpha^{(t)}\|$ . By the triangle inequality

$$\begin{aligned} & \left| \|\bar{\alpha}^{(t+1)} - \alpha^{(t)}\| - \|\alpha^{(t+1)} - \alpha^{(t)}\| \right| \\ & \leq \|\bar{\alpha}^{(t+1)} - \alpha^{(t+1)}\| \\ & \leq \left\| \Pi_{\mathcal{A}}\left(\alpha^{(t)} - \eta_\alpha \nabla_\alpha \mathcal{F}_\gamma(\alpha^{(t)}, w_K^{(t)}, u_K^{(t)})\right) - \Pi_{\mathcal{A}}\left(\alpha^{(t)} - \eta_\alpha \nabla_\alpha \mathcal{H}_\gamma(\alpha^{(t)}, w_\gamma^*(\alpha^{(t)}))\right) \right\| \\ & \leq \eta_\alpha \left\| \nabla_\alpha \mathcal{F}_\gamma(\alpha^{(t)}, w_K^{(t)}, u_K^{(t)}) - \nabla_\alpha \mathcal{H}_\gamma(\alpha^{(t)}, w_\gamma^*(\alpha^{(t)})) \right\| \\ & \leq \eta_\alpha \gamma B \|w_K^{(t)} - w_\gamma^*(\alpha^{(t)})\| + \eta_\alpha \gamma B \|u_K^{(t)} - w^*(\alpha^{(t)})\|, \end{aligned} \quad (27)$$

where the last inequality is from the smoothness Assumption 2.  $w_\gamma^*(\alpha^{(t)})$  and  $w^*(\alpha^{(t)})$  are respectively the projection of  $w_K^{(t)}$  and  $u_K^{(t)}$  to  $S_\gamma^*(\alpha^{(t)})$  and  $S^*(\alpha^{(t)})$ . Firstly, due to the PL-condition and Lipschitz smoothness of  $\mathcal{L}_{\text{train}}$ , we have

$$\begin{aligned} \mathcal{L}_{\text{train}}(\alpha^{(t)}, u_{k+1}^{(t)}) - \mathcal{L}_{\text{train}}(\alpha^{(t)}, u_k^{(t)}) & \leq \left\langle \nabla_w \mathcal{L}_{\text{train}}(\alpha^{(t)}, u_k^{(t)}), u_{k+1}^{(t)} - u_k^{(t)} \right\rangle + \frac{L}{2} \|u_{k+1}^{(t)} - u_k^{(t)}\|^2 \\ & = -\frac{1}{2L} \|\nabla_w \mathcal{L}_{\text{train}}(\alpha^{(t)}, u_k^{(t)})\|^2 \\ & \leq -\frac{\mu}{L} \left( \mathcal{L}_{\text{train}}(\alpha^{(t)}, u_k^{(t)}) - \mathcal{L}_{\text{train}}(\alpha^{(t)}, w^*(\alpha^{(t)})) \right), \end{aligned} \quad (28)$$

which implies

$$\begin{aligned} \|u_K^{(t)} - w^*(\alpha^{(t)})\|^2 & \leq \frac{2}{\mu} \left(1 - \frac{\mu}{L}\right)^{K+E} \left( \mathcal{L}_{\text{train}}(\alpha^{(t)}, u_0^{(t)}) - \mathcal{L}_{\text{train}}(\alpha^{(t)}, w^*(\alpha^{(t)})) \right) \\ & \leq \frac{2}{\mu} \left(1 - \frac{\mu}{L}\right)^{K+E} D \\ & = \mathcal{O}\left(T^{-\frac{3}{4}}\right). \end{aligned} \quad (29)$$

557 due to the selection of  $K$  and  $E$  and the PL condition of  $\mathcal{L}_{\text{train}}(\boldsymbol{\alpha}, \mathbf{w})$  (which implies the error bound  
 558 condition [15]). Similarly, we can prove that

$$\begin{aligned}\|\mathbf{w}_K^{(t)} - \mathbf{w}_\gamma^*(\boldsymbol{\alpha}^{(t)})\|^2 &\leq \frac{2}{\mu_\gamma} \left(1 - \frac{\mu}{L}\right)^K \left(\mathcal{H}_\gamma(\boldsymbol{\alpha}^{(t)}, \mathbf{w}_K^{(t)}) - \mathcal{H}_\gamma(\boldsymbol{\alpha}^{(t)}, \mathbf{w}_\gamma^*(\boldsymbol{\alpha}^{(t)}))\right) \\ &\leq \frac{2}{\mu_\gamma} \left(1 - \frac{\mu_\gamma}{L_\gamma}\right)^K (1 + \gamma) D \\ &= \mathcal{O}\left(T^{-\frac{3}{4}}\right).\end{aligned}\tag{30}$$

559 Combining (26) (27), (29), and (30), we have

$$\mathcal{H}_\gamma(\boldsymbol{\alpha}^{(t+1)}, \mathbf{w}_\gamma^*(\boldsymbol{\alpha}^{(t+1)})) - \mathcal{H}_\gamma(\boldsymbol{\alpha}^{(t)}, \mathbf{w}_\gamma^*(\boldsymbol{\alpha}^{(t)})) \leq -\frac{1}{4\eta_\alpha} \|\boldsymbol{\alpha}^{(t+1)} - \boldsymbol{\alpha}^{(t)}\|^2 + \mathcal{O}\left(\frac{\eta_\alpha \gamma^2}{T^{\frac{3}{2}}}\right), \tag{31}$$

560 so that, by combining (27), we get

$$\begin{aligned}\frac{1}{T} \sum_{t=1}^T \frac{1}{\eta_\alpha^2} \|\bar{\boldsymbol{\alpha}}^{(t+1)} - \boldsymbol{\alpha}^{(t)}\|^2 &\leq \frac{1}{T} \sum_{t=1}^T \frac{2}{\eta_\alpha^2} \left[ \|\boldsymbol{\alpha}^{(t+1)} - \boldsymbol{\alpha}^{(t)}\|^2 + \|\bar{\boldsymbol{\alpha}}^{(t+1)} - \boldsymbol{\alpha}^{(t)}\| - \|\boldsymbol{\alpha}^{(t+1)} - \boldsymbol{\alpha}^{(t)}\|^2 \right] \\ &\leq \frac{\mathcal{H}_\gamma(\boldsymbol{\alpha}^{(0)}, \mathbf{w}_\gamma^*(\boldsymbol{\alpha}^{(0)})) - \inf_{\boldsymbol{\alpha}} \mathcal{H}_\gamma(\boldsymbol{\alpha}^{(0)}, \mathbf{w}_\gamma^*(\boldsymbol{\alpha}^{(0)}))}{\eta_\alpha T} + \mathcal{O}\left(\frac{\gamma^2}{T^{\frac{3}{2}}}\right) \\ &= \mathcal{O}\left(\frac{1}{\sqrt{T}}\right).\end{aligned}\tag{32}$$

561 Due to the definition of  $\bar{\boldsymbol{\alpha}}^{(t+1)}$ , combining this with (24), and the value of  $\gamma$  proves our conclusion.  
 562  $\square$

## 563 B Proof of the Proposition

564 **Proposition 1.** Assume  $\mathcal{L}_{\text{train}}^m = \mathcal{L}_{\text{val}}^m$ , the uniform mixture ratio  $\bar{\alpha}_m = \frac{1}{M}$  for  $m = 1, \dots, M$   
 565 constitutes a valid solution of (1).

566 *Proof.* Let  $\mathcal{L}_{\text{train}}^m = \mathcal{L}_{\text{val}}^m = \mathbb{E}_{\text{data}}[\mathcal{L}_{\text{data}}^m] := \mathcal{L}^m$ . To establish the above, it suffices to show:

$$\sum_{m=1}^M \mathcal{L}_{\text{val}}^m(\mathbf{w}^*(\bar{\boldsymbol{\alpha}})) \leq \sum_{m=1}^M \mathcal{L}_{\text{val}}^m(\mathbf{w}^*(\boldsymbol{\alpha}))$$

567 which is equivalent to:

$$\sum_{m=1}^M \bar{\alpha}_m \mathcal{L}^m(\mathbf{w}^*(\bar{\boldsymbol{\alpha}})) \leq \sum_{m=1}^M \bar{\alpha}_m \mathcal{L}^m(\mathbf{w}^*(\boldsymbol{\alpha})) \tag{33}$$

568 Because  $\mathbf{w}^*(\bar{\boldsymbol{\alpha}})$  is the minimizer of the inner-level problem under uniform weighting, we have:

$$\sum_{m=1}^M \bar{\alpha}_m \mathcal{L}^m(\mathbf{w}^*(\bar{\boldsymbol{\alpha}})) \leq \sum_{m=1}^M \bar{\alpha}_m \mathcal{L}^m(\mathbf{w}) \tag{34}$$

569 for any  $\mathbf{w}$ . In particular, by choosing  $\mathbf{w} = \mathbf{w}^*(\boldsymbol{\alpha})$ , the right-hand side becomes the expression  
 570 in (33), completing the proof.  $\square$

## 571 C Implementations and Hyper-parameters

572 **Hyper-parameter settings** The detailed hyper-parameters for the TANDEM algorithms in the  
 573 conventional data-abundant pretraining scenario, the data-restricted training scenario, and supervised  
 574 fine-tuning are shown in Table 5. Given the optimized mixture ratio, we resample the corpus to  
 575 construct the final dataset for model training.

Table 5: Hyper-parameters of TANDEM for different application scenarios

	Data Audent	Data Restricted		Supervised Fine-tuning	
	GPT-like 160M	160M	410M	1B	Qwen2-0.5B
Batch Size	8	128	128	128	32
Learning Rate $\eta_u$	5e-5	5e-4	5e-4	5e-4	4e-6
Learning Rate $\eta_w$	5e-5	5e-4	5e-4	5e-4	4e-6
Learning Rate $\eta_\alpha$	2e-3	4e-3	4e-3	4e-3	4e-3
Learning Rate Scheduler	Cosine	Cosine	Cosine	Cosine	Cosine
Penalty $\gamma$	1.0	1.0	1.0	1.0	1.0
Probing Steps K	5	5	5	5	20
Free Training Steps E	20	5	5	5	10
Total Steps (w.r.t $u$ )	40000	5000	5000	5000	2000
Context Length	2048	512	512	512	512
Weight Decay	1e-2	1e-2	1e-2	1e-2	1e-2
Gradient Clipping	1.0	1.0	1.0	1.0	1.0

Table 6: Comparison of models of different sizes in the data-abundant pretraining scenario and data restricted scenario with standard deviation.

	Data Audent Regime		Data Restricted Regime	
	160M Avg.	160M Avg.	410M Avg.	1B Avg.
Uniform	25.74 $\pm$ 0.13	31.53 $\pm$ 0.11	29.59 $\pm$ 0.02	29.91 $\pm$ 0.09
DoReMi	28.32 $\pm$ 0.12	36.91 $\pm$ 0.09	54.61 $\pm$ 0.60	56.53 $\pm$ 0.53
DoGE	26.60 $\pm$ 0.21	30.10 $\pm$ 0.05	27.45 $\pm$ 0.02	-
Skill-It	25.87 $\pm$ 0.07	29.24 $\pm$ 0.08	27.70 $\pm$ 0.03	27.15 $\pm$ 0.17
Aioli	25.66 $\pm$ 0.14	31.19 $\pm$ 0.25	28.79 $\pm$ 0.06	28.07 $\pm$ 0.03
TANDEM	<b>25.43</b> $\pm$ 0.15	<b>28.07</b> $\pm$ 0.07	<b>25.00</b> $\pm$ 0.01	<b>24.35</b> $\pm$ 0.03

## D Comparison with Standard Deviation

We test each method in Table 2, Table 3 and Table 4 3 times. The average accuracy and standard deviation are reported in Table 6 and Table 7.

## E Visualization of the Optimized Mixture Ratios.

We visualize the mixture ratio learned in each application scenarios in Figure 9, Figure 10 and Figure 11. For the baselines, the average proportion over the entire training trajectory is taken. While for TANDEM, we use the average mixture ratio at the last 10% training trajectory.

Table 7: Comparison for the 500M Qwen-2 model on a mixture of 6 supervised fine-tuning tasks with standard deviation.

Method	SQuAD1.1	AMRSum	MuTual	SemEval	SST2	BoolQ	Avg. Metric $\uparrow$
Uniform	72.62 $\pm$ 0.28	45.18 $\pm$ 0.19	72.72 $\pm$ 0.58	<b>89.75</b> $\pm$ 0.58	87.75 $\pm$ 0.20	80.29 $\pm$ 0.60	74.72 $\pm$ 0.18
DoReMi	71.40 $\pm$ 0.71	43.40 $\pm$ 0.08	70.97 $\pm$ 0.57	88.50 $\pm$ 0.44	87.00 $\pm$ 0.45	77.43 $\pm$ 0.16	73.11 $\pm$ 0.18
DoGE	71.26 $\pm$ 0.44	44.81 $\pm$ 0.34	71.67 $\pm$ 0.16	89.00 $\pm$ 0.16	88.30 $\pm$ 0.45	<b>81.04</b> $\pm$ 0.16	74.35 $\pm$ 0.18
Skill-It	72.14 $\pm$ 0.13	44.21 $\pm$ 0.37	73.07 $\pm$ 0.51	89.40 $\pm$ 0.14	88.40 $\pm$ 0.75	80.34 $\pm$ 0.22	74.60 $\pm$ 0.15
Aioli	72.35 $\pm$ 0.42	45.01 $\pm$ 0.18	73.57 $\pm$ 0.68	89.10 $\pm$ 0.32	88.10 $\pm$ 0.12	79.64 $\pm$ 0.41	74.63 $\pm$ 0.15
TANDEM	<b>72.73</b> $\pm$ 0.19	<b>45.19</b> $\pm$ 0.04	<b>73.77</b> $\pm$ 0.44	89.70 $\pm$ 0.30	<b>88.70</b> $\pm$ 0.60	80.04 $\pm$ 0.70	<b>75.03</b> $\pm$ 0.14

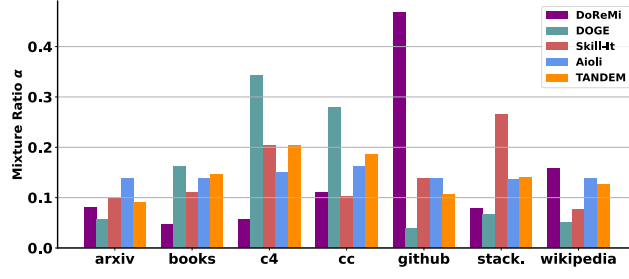
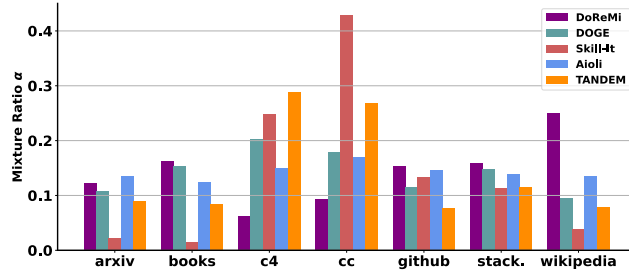
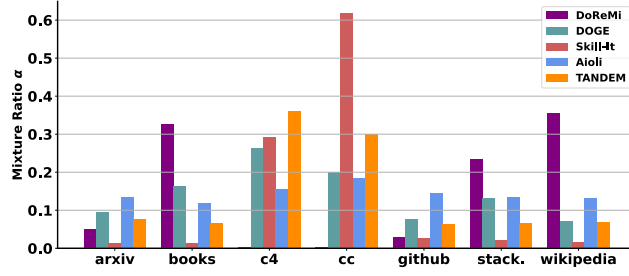


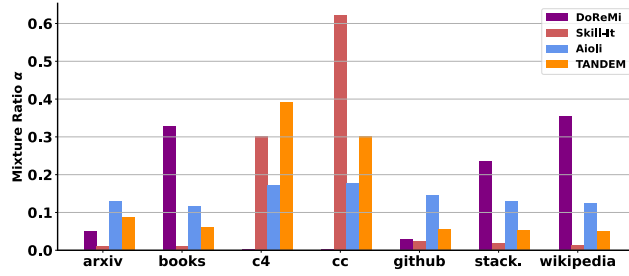
Figure 9: Mixture ratio learned by different methods in the data-abundant pretraining.



(a) 160M Model



(b) 410M Model



(c) 1B Model

Figure 10: Mixture ratio learned by different methods in the supervised fine-tuning.



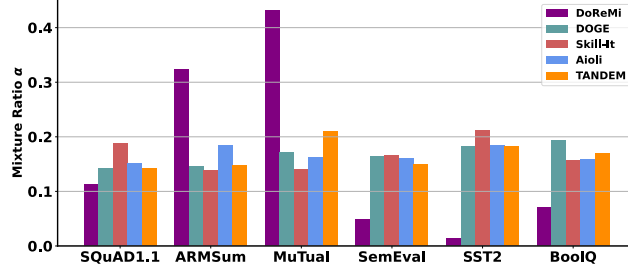


Figure 11: Mixture ratio learned by different methods in the data-abundant pretraining.

## 583 F Summary of Notations

Table 8: Summary of the notations used throughout this paper. Variables only used in theoretical analysis are grayed for better readability.

Topic	Notation	Explanation
Data Sets	$M$	The number of domains.
	$\mathcal{D}_m$	The $m$ -th domain data.
	$\mathcal{D}_{\text{train}}$	Train set.
	$\mathcal{D}_{\text{val}}$	Validation set.
Models & Parameters	$\mathbf{u}$	Parameters of the proxy model.
	$\mathbf{w}$	Parameters of the reference model.
	$\mathbf{w}^*$	Optimal solution for the lower level problem.
	$\mathbf{w}_\gamma^*$	Optimal solution for the penalized problem.
	$\mathcal{S}^*(\alpha)$	Solution set for the lower level problem.
	$\mathcal{S}_\gamma^*(\alpha)$	Solution set for the penalized problem.
	$\alpha$	Data mixture ratio
Problems & Losses	$\mathcal{A}$	The probability simplex.
	$\mathcal{L}_{\text{train}}^m$	Train loss on the $m$ -th domain.
	$\mathcal{L}_{\text{val}}^m$	Validation loss on the $m$ -th domain.
	$\mathcal{L}_{\text{train}}$	Overall train loss weighted by the mixture ratio $\alpha$ .
	$\mathcal{L}_{\text{val}}$	Overall validation loss.
	$\mathcal{H}(\alpha)$	The upper-level loss with the lower-level problem optimized.
	$\mathcal{H}_\gamma(\alpha, \mathbf{w})$	The loss of the penalized problem.
Function Properties	$\mu$	PL coefficient of the lower-level problem.
	$\mu_\gamma$	PL coefficient of the penalized problem.
	$L$	Lipschitz constant for $\nabla_{\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$ on $\mathbf{w}$ .
	$B$	Lipschitz constant for $\mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$ and $\mathcal{L}_{\text{val}}(\mathbf{w})$ on $\mathbf{w}$ .
	$\lambda$ and $\rho$	Hessian $\nabla_{\mathbf{w}\mathbf{w}} \mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) \succeq \lambda$ and $\nabla_{\alpha\mathbf{w}}^2 \mathcal{L}_{\text{train}}(\alpha, \mathbf{w}) \preceq \rho$
	$H$	Lipschitz constant for $\nabla_{\alpha\mathbf{w}}^2 \mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$ and $\nabla_{\mathbf{w}\mathbf{w}}^2 \mathcal{L}_{\text{train}}(\alpha, \mathbf{w})$ .
	$D$	Upper bound for the train/validation loss.
	$L_\gamma$	Lipschitz constant for $\nabla_{\alpha} \mathcal{H}_\gamma(\alpha, \mathbf{w}_\gamma^*(\alpha))$ .
Train	$t$	Mixture ratio $\alpha$ training step
	$T$	The total number of $\alpha$ update.
	$k$	$\mathbf{u}$ and $\mathbf{w}$ update step.
	$e$	Free $\mathbf{u}$ update step.
	$K$	The number of $\mathbf{u}, \mathbf{w}$ probing update for one $\alpha$ update.
	$E$	The number of $\mathbf{u}$ free update.
	$\eta_\alpha$	The learning rate on $\alpha$ .
	$\eta_{\mathbf{u}}$	The learning rate on $\mathbf{u}$ .
	$\eta_{\mathbf{w}}$	The learning rate on $\mathbf{w}$ .
	$\gamma$	The penalty strength.

## 584 **G Limitations and Future Works**

585 Despite the advancements introduced in this work, several challenges remain open for future research.  
586 The limitations of this paper are as follows: (1) Due to limited computational resources, our ex-  
587 periments were conducted using models of up to 1 billion parameters. Although our experimental  
588 results demonstrate TANDEM’s effectiveness at this scale, the constraint on model size limits our  
589 ability to verify whether our findings generalize to significantly larger models, such as those with 405  
590 billion parameters. (2) Our current data mixture optimization (DMO) approach is validated on coarse,  
591 naturally occurring domain splits. For example, the SlimPajama corpus consists of seven domains  
592 sourced from different origins. The impact of using more fine-grained and intentionally designed  
593 domain splits on DMO performance remains unexplored and presents an interesting direction for  
594 future investigation.