# PiKE: Adaptive Data Mixing for Multi-Task Learning Under Low Gradient Conflicts

**Anonymous authors**
Paper under double-blind review

## Abstract

Modern machine learning models are trained on diverse datasets and tasks to improve generalization. A key challenge in multitask learning is determining the optimal data mixing and sampling strategy across different data sources. Prior research in this multi-task learning setting has primarily focused on mitigating gradient conflicts between tasks. However, we observe that many real-world multitask learning scenarios—such as multilingual training and multi-domain learning in large foundation models—exhibit predominantly positive task interactions with minimal or no gradient conflict. Building on this insight, we introduce PiKE (**P**ositive gradient **i**nteraction-based **K**-task weights **E**stimator), an adaptive data mixing algorithm that dynamically adjusts task contributions throughout training. PiKE optimizes task sampling to minimize overall loss, effectively leveraging positive gradient interactions with almost no additional computational overhead. We establish theoretical convergence guarantees for PiKE and demonstrate its superiority over static and non-adaptive mixing strategies. Additionally, we extend PiKE to promote fair learning across tasks, ensuring balanced progress and preventing task underrepresentation. Empirical evaluations on large-scale language model pretraining show that PiKE consistently outperforms existing heuristic and static mixing strategies, leading to faster convergence and improved downstream task performance.

## 1 Introduction

Modern foundation models, such as large language models (LLMs), have demonstrated impressive generalization and multitask learning capabilities by pretraining on diverse datasets across multiple domains (Liu et al., 2024a; Team et al., 2024a; Chowdhery et al., 2022; Radford et al., 2019). The effectiveness of these models is heavily influenced by the composition of their training data (Du et al., 2022; Hoffmann et al., 2022). However, determining the optimal data mixture (across different tasks and data sources) remains a fundamental challenge due to the substantial size of both models and datasets, as well as the high computational cost of training. In most cases, training large models is limited to a single experimental run, making it impractical to iteratively fine-tune the weights of different data sources/tasks.

Current approaches to multitask learning typically rely on fixed dataset weights (aka mixing or sampling strategies), often determined heuristically or based on the performance of smaller proxy models. For example, mT5 (Xue, 2020) assigns dataset weights based on their relative abundance, GLaM (Du et al., 2022) selects weights by evaluating downstream performance on smaller models, and the 405B LLaMA-3 model (Dubey et al., 2024) heuristically constructs its training corpus from diverse sources. More recently, DoReMi (Xie et al., 2024) introduced a method that pretrains a small model using group distributionally robust optimization to determine dataset weights for larger-scale training. However, the optimality of these approaches is unclear, as the capabilities of large and small models differ significantly (Team et al., 2024b; Wortsman et al., 2023). Moreover, the loss landscape evolves throughout training (Zhang et al., 2024; Li et al., 2018), meaning that static dataset weights determined at initialization may not remain optimal (as we will further elaborate in Section 3.1).

Another line of research addresses multitask optimization by modifying gradient updates to mitigate gradient conflicts, where task gradients point in opposing directions, slowing down optimization. Techniques such as PCGrad (Yu et al., 2020), GradNorm (Chen et al., 2018), and MGDA (Désidéri, 2012) attempt to minimize these conflicts by adjusting gradient directions during training. While these methods improve performance, they introduce significant computational and memory overhead, making them impractical for large-scale models with numerous tasks (Xin et al., 2022). Furthermore, while gradient conflicts are prevalent in vision-based multitask learning (Wang et al., 2020; Liu et al., 2021) and small-scale language models, we observe that they rarely occur when training large language models, as we will elaborate in Section 3. Instead, task gradients in such models often exhibit positive interactions, suggesting that existing conflict-mitigation strategies may not be necessary for large-scale multitask learning. Given these observations, we pose the following question:

*Can we design a multitask learning mixing strategy that leverages the absence of gradient conflict to improve efficiency and performance in training large models on diverse datasets?*

To answer this, we introduce PiKE (Positive gradient interaction-based K-task weight Estimator), a novel *adaptive* data mixing strategy that dynamically adjusts task contributions throughout training. Unlike static and heuristic approaches, PiKE optimizes data allocation based on gradient information, effectively leveraging positive gradient interactions to enhance model performance. Our key contributions are as follows:

1. We propose PiKE, an approach that dynamically adjusts the mixture of data sources during training based on task gradient magnitudes and variance. This enables PiKE to scale efficiently with increasing model size and the number of tasks, overcoming the limitations of static and heuristic task weighting strategies.
2. We establish the theoretical convergence of PiKE when applied with stochastic gradient descent (SGD). Additionally, we extend PiKE to incorporate tilted empirical risk minimization (Li et al., 2020; Mo & Walrand, 2000), promoting fair learning across tasks and preventing task underrepresentation.
3. We conduct comprehensive experiments across various language multitask learning settings, including pretraining language models on multilingual text corpora and English datasets from diverse domains. Across different scales (110M, 270M, 750M, and 1B parameters) and scenarios, PiKE consistently outperforms existing static and heuristic data mixing methods. Notably, in multilingual pretraining for 1B models, PiKE improves average downstream accuracy by $7.1\%$ and achieves baseline accuracy $1.9\times$ faster. On the GLaM dataset with 750M models, PiKE surpasses DoReMi (Xie et al., 2024) by $3.4\%$. Importantly, PiKE achieves these improvements with only negligible additional computational overhead.

The rest of this paper is structured as follows. Section 2 introduces notations and problem formulation. Section 3 presents the PiKE algorithm, its theoretical analysis, and an extension for fairness. Section 4 provides experimental results, followed by discussions in Section 5. Further related work is discussed in Appendix A.

# 2 Preliminaries

## 2.1 Problem Definition and Notations

We aim to train a *single* model with parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ to perform $K \geq 2$ tasks simultaneously. Each task is associated with a smooth (possibly non-convex) loss function $\ell_k(\boldsymbol{\theta}, x) : \mathbb{R}^d \times \mathbb{R}^{d_x} \to \mathbb{R}$ where $x$ is the data point. Then, it is common to minimize the total expected loss:

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \mathcal{L}(\boldsymbol{\theta}) := \sum_{k=1}^{K} \mathbb{E}_{x \sim \mathcal{D}_k}[\ell_k(\boldsymbol{\theta}; x)], \tag{1}$$

where $\mathcal{D}_k$ represents the data distribution for task $k$. We define $\mathcal{L}_k(\boldsymbol{\theta}) := \mathbb{E}_{x \sim \mathcal{D}_k}[\ell_k(\boldsymbol{\theta}; x)]$. For notation, $\|\cdot\|$ represents the Euclidean norm, $\mathrm{Tr}(\cdot)$ denotes the trace operator, and a function $h$ is $L$-Lipschitz if $\|h(\boldsymbol{\theta}) - h(\boldsymbol{\theta}')\| \leq L\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|$ for any $\boldsymbol{\theta}, \boldsymbol{\theta}'$ in the domain of $h(\cdot)$. A function $f(\cdot)$ is $L$-smooth if its gradient is $L$-Lipschitz continuous.

## 2.2 Sampling Strategies: Random, Round-Robin, and Mix

To optimize equation (1) using stochastic optimizers such as Adam or SGD, we must select batches from one or multiple tasks at each training step. The choice of batch selection strategy significantly

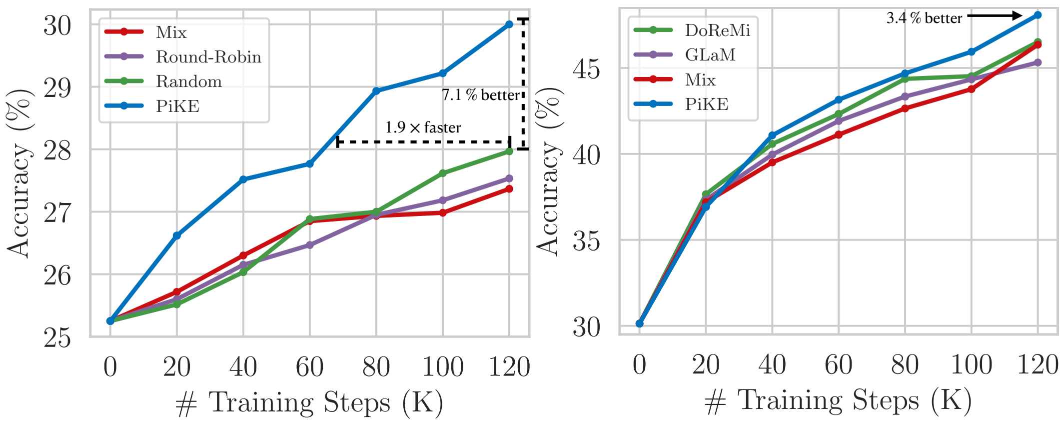



Figure 1: **Pre-training metric (average downstream task accuracy)**, higher is better. **Left:** 1B models on multilingual C4 (en) and C4 (hi) datasets. **Right:** 750M models on GLaM datasets with six domains. PiKE dynamically optimizes $K$-task weights during language model pre-training. We compare PiKE against baselines in two multitask learning scenarios: multilingual training and the training on GLaM dataset. Mix uses equal batch size for each task ($b_k = b/K, \forall k \in K$), GLaM Du et al. (2022) uses fixed domain weights tuned for downstream performance, and DoReMi Xie et al. (2024) requires pre-training a smaller model to determine optimized weights for training larger models. PiKE introduces negligible computation and memory overhead while outperforming all baselines. In pre-training 1B language models on multilingual C4 (en) and C4 (hi), PiKE improves average downstream accuracy by 7.1% and achieves baseline accuracy $1.9\times$ faster. For 750M models pre-trained on the GLaM dataset, PiKE improves average downstream accuracy by 3.4% compared to DoReMi. Tables 8 and 9 provide additional experiments and detailed results.

impacts model performance (Bengio et al., 2009; Ge et al., 2024; Ye et al., 2024; Xie et al., 2024; Liu et al., 2024c). Below, we define three common sampling strategies: *Random*, *Round-Robin*, and *Mix*.

**Random Sampling.** At each step, a single task $k$ is randomly chosen with probability $p_k$ ($\sum_{k=1}^{K} p_k = 1$), and a batch of $b$ samples is drawn from $\mathcal{D}_k$ (dataset of task $k$). The model parameters $\boldsymbol{\theta}$ are updated using the gradient of the selected task's loss function evaluated on the batch.

**Round-Robin Sampling.** Tasks are selected cyclically, ensuring each task is chosen once every $K$ steps. At iteration $t$, task $k = (t \bmod K) + 1$ is selected, and a batch is sampled from $\mathcal{D}_k$. The model parameters are then updated based on the loss gradient evaluated on the selected batch.

**Mix Sampling.** Each batch contains samples from all $K$ tasks, with $b_k$ samples drawn from $\mathcal{D}_k$ such that the total batch size is $b = \sum_{k=1}^{K} b_k$. The model update at iteration $t$ is based on the combined gradient:

$$\mathbf{g}_t = \frac{1}{b} \sum_{k=1}^{K} \sum_{i=1}^{b_k} \nabla \ell_k(\boldsymbol{\theta}_t; x_i),\ x_i \sim \mathcal{D}_k. \tag{2}$$

Unlike the Random and Round-Robin, Mix strategy ensures that each task contributes to the computed gradient at each optimization step.

Historically, *Mix* has been preferred in computer vision multitask learning (Dai et al., 2016; Misra et al., 2016; Chen et al., 2018; Ruder et al., 2019; Yu et al., 2020; Liu et al., 2024b), while *Random* and *Round-Robin* have been more common in early language model multitask training (Liu et al., 2015; Luong et al., 2015; Liu et al., 2019). Recent studies on large-scale language models (Devlin, 2018; Raffel et al., 2020; Brown et al., 2020; Team et al., 2023) have revisited these strategies, finding that **Mix** generally yields superior performance, particularly when training across diverse datasets (Du et al., 2022; Chowdhery et al., 2023; Xie et al., 2024; Raffel et al., 2020; Gao et al., 2020; Wang et al., 2019). Figure 2 (and Figure 4 in the appendix) illustrate this by comparing downstream accuracy on multilingual mC4 (Xue, 2020) and GLaM (Du et al., 2022) datasets. Across all scenarios, *Mix* consistently outperforms the other two strategies, motivating its use in pretraining large language models.

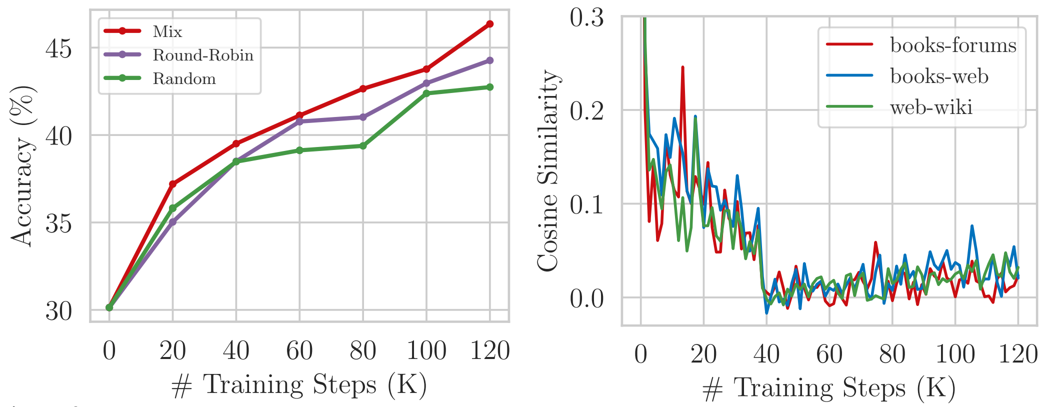



Figure 2: **Left:** Average accuracy across four downstream tasks (ArcE, CSQA, HellaSwag, and PIQA) for 750M GPT-2 large-style language models pre-trained using Mix, Round-Robin, and Random sampling strategies. Mix allocates equal batch sizes ($b_k = b/K, \forall k \in K$), while Random employs uniform sampling ($p_k = 1/K,\ \forall k$). Additional results are available in Appendix E.1. **Right:** Cosine similarity between task gradients during pre-training 750M GPT-2 style language model on GLaM datasets. "*data1-data2*" denotes the cosine similarity between the gradient evaluated on *data1* (task 1) and the gradient of *data2* (task 2). More results can be found in Appendix E.2.

### 2.3 Gradient Conflicts in Multitask Learning

A key challenge in multitask learning prior literature is managing *gradient conflicts* (Liu et al., 2021; Yu et al., 2020), where the gradient of a task opposes the overall optimization direction. Formally, a conflict occurs at iteration $t$ if there exists a task $k$ such that

$$\langle \nabla \mathcal{L}(\boldsymbol{\theta}_t), \nabla \mathcal{L}_k(\boldsymbol{\theta}_t) \rangle < 0,$$

indicating that updating $\boldsymbol{\theta}_t$ may increase the loss for task $k$, thereby hindering balanced learning across tasks. Existing methods attempt to mitigate gradient conflicts by adjusting gradients (Yu et al., 2020), but these approaches introduce computational overhead, often requiring $\mathcal{O}(K)$ complexity per step, making them impractical for large-scale models.

While gradient conflicts are common in vision-based multitask learning and small-scale language models, we observe that they rarely occur in large-scale language model training. **In such models, task gradients are typically aligned (or close to orthogonal) rather than conflicting.** This insight suggests that instead of mitigating conflicts, a more effective strategy is to leverage nonnegative gradient interactions to enhance training efficiency—a key motivation for our approach, as discussed in the next section.

## 3 Method

### 3.1 Motivation

Our approach is based on two key observations: (1) gradient conflicts are rare in LLMs, and (2) the Mix sampling strategy can be made adaptive rather than static:

#### 3.1.1 Reevaluating Gradient Conflicts in LLMs

The assumption that gradient conflicts dominate multitask learning does not necessarily hold for LLM pretraining. Our experiments show that task gradients in such models exhibit minimal conflicts. To illustrate this, we pretrain (i) a 1B GPT-2-style (Radford et al., 2019) model on the multilingual mC4 dataset (Xue, 2020) (six languages: English, Hindi, German, Chinese, French, and Arabic) and (ii) a 750M model on the GLaM dataset (Du et al., 2022) (English text from six domains). Experimental details are in Appendix D. Figures 2 and 5 show cosine similarity trends for task gradients. Key observations are: 1) Gradient similarity starts high but decreases over time. 2) Multilingual gradient similarity varies with linguistic proximity (e.g., English-German align closely), while GLaM tasks exhibit uniformly aligned gradients. 3) Task gradients rarely conflict—multilingual cosine similarity seldom drops below $-0.1$, while GLaM gradients remain mostly positive. These patterns align with prior work (Wang et al., 2020).

These findings challenge the conventional focus on mitigating gradient conflicts in multitask learning. Therefore, **instead of reducing conflicts, we should leverage non-conflicting gradients**. Existing conflict-aware methods like PCGrad (Yu et al., 2020) and AdaTask (Yang et al., 2023) are ineffective in this setting since they focus on resolving gradient conflict (which is indeed not present). As shown in Figure 7, 1) PCGrad performs similarly to Mix, as it only adjusts gradients when conflicts occur—which is rare. 2) AdaTask converges slower due to noisy gradients and suboptimal optimizer state updates. Additionally, both methods are memory-intensive, requiring $O(K)$ storage for task gradients (PCGrad) or optimizer states (AdaTask), making them impractical for large models like the 540B PaLM (Chowdhery et al., 2022).

Crucially, these methods fail to exploit the *non-conflicting* interactions among tasks, focusing instead on resolving conflicts that seldom arise. This highlights the need for a new approach that actively leverages lack of gradient conflicts to enhance training efficiency.

### 3.1.2 Adaptive versus Static Mixing

Prior work using the Mix sampling strategy typically relies on fixed (static) sampling weights, keeping $(b_1, \ldots, b_K)$ constant throughout training. However, dynamically adjusting batch composition can significantly enhance efficiency. We illustrate this with a simple example:

**Example 3.1.** Consider training on $K = 2$ tasks with losses $\ell_1(\boldsymbol{\theta}; x_1) = \frac{1}{2}(\boldsymbol{\theta}^\top e_1)^2 + x_1^\top \boldsymbol{\theta}$ and $\ell_2(\boldsymbol{\theta}; x_2) = \frac{1}{2}(\boldsymbol{\theta}^\top e_2)^2 + x_2^\top \boldsymbol{\theta}$, where $e_1 = [1\ 0]^\top$, $e_2 = [0\ 1]^\top$, and $\boldsymbol{\theta} \in \mathbb{R}^2$. Data for task 1 follows $x_1 \sim \mathcal{N}(0, \sigma_1^2 I)$, while task 2 follows $x_2 \sim \mathcal{N}(0, \sigma_2^2 I)$. The overall loss for task $k$ simplifies to $\mathcal{L}_k(\boldsymbol{\theta}) = \frac{1}{2}(\boldsymbol{\theta}^\top e_k)^2$. Using $b_1$ samples from task 1 and $b_2$ samples from task 2 in a batch at iteration $t$, the gradient is:

$$\mathbf{g}_t = \frac{1}{b_1 + b_2} \left( b_1 e_1 e_1^\top + b_2 e_2 e_2^\top \right) \boldsymbol{\theta}_t + \mathbf{z},$$

where $\mathbf{z} \sim \mathcal{N}(0, \frac{b_1\sigma_1^2 + b_2\sigma_2^2}{b^2} I)$ with $b = b_1 + b_2$. Updating $\boldsymbol{\theta}_t$ via SGD, $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \mathbf{g}_t$, we have

$$\begin{aligned}\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_{t+1})] = &\frac{1}{2}(1 - \eta\frac{b_1}{b})^2\theta_{1,t}^2 + \frac{1}{2}(1 - \eta\frac{b_2}{b})^2\theta_{2,t}^2 \\ &+ \eta^2 \frac{b_1\sigma_1^2 + b_2\sigma_2^2}{b^2},\end{aligned} \tag{3}$$

where $\theta_{1,t}$ and $\theta_{2,t}$ denote the first and second component of the vector $\boldsymbol{\theta}_t$. The derivation details of equation (3) can be found in Appendix F.1. Letting $w_1 := \frac{b_1}{b}$, $w_2 := \frac{b_2}{b}$, and relaxing them to take real values, we can optimize the mixing weights $w_1$ and $w_2$ as

$$w_1^* = \Pi \left( \frac{b^{-1}(\sigma_2^2 - \sigma_1^2) + \eta^{-1}(\theta_{1,t}^2 - \theta_{2,t}^2) + \theta_{2,t}^2}{\theta_{1,t}^2 + \theta_{2,t}^2} \right) \tag{4}$$

and $w_2^* = 1 - w_1^*$ where $\Pi(\xi) = \min\{\max\{\xi, 0\}, 1\}$ is the projection operator onto the interval $[0, 1]$. This result shows that optimal batch composition $b_1, b_2$ should evolve over time to maximize training efficiency.

Figure 3 compares static mixing strategies with an adaptive approach based on equation (4), highlighting the superiority of adaptive mixing. Moreover, the adaptive mixing strategy in this example does not require any hyperparameter tuning, while finding the best static mixing requires tuning. This simple example mirrors key aspects of multitask learning in large models: 1) The optimal solution $\boldsymbol{\theta}^* = 0$ minimizes all task losses simultaneously, reflecting the high expressive power of large models. 2) Task gradients are non-conflicting, resembling real-world gradient interactions observed in Figure 2. Moreover, equation (4) further reveals that optimal data mixing depends on (1) gradient norm squared per task ($\|\nabla\mathcal{L}_1(\boldsymbol{\theta})\|^2 = \theta_1^2$, $\|\nabla\mathcal{L}_2(\boldsymbol{\theta})\|^2 = \theta_2^2$) and (2) gradient variance ($\sigma_1^2$, $\sigma_2^2$). As we will see next, these factors play a crucial role in defining optimal mixing strategies for more general settings.

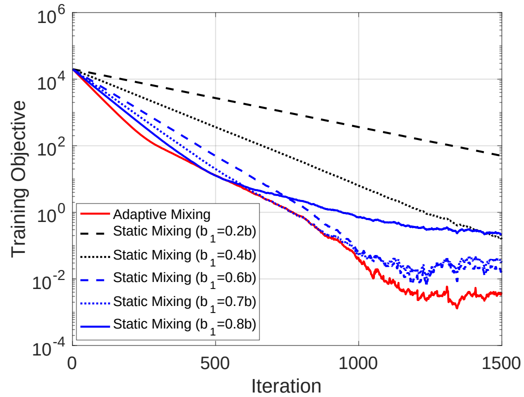



Figure 3: Adaptive vs. static mixing for Example 3.1. Adaptive mixing consistently outperforms static mixing.

### 3.2 PiKE: Conceptual Version

As discussed in Section 2, Mix sampling provides greater stability and generalization than Random and Round-Robin in LLM pretraining. Therefore, we focus on Mix but adopt a dynamic rather than static approach, as motivated in Section 3.1. To develop our method and motivated by the discussions in section 3.1, we first quantify gradient conflicts:

**Definition 3.2.** For a given point $\boldsymbol{\theta}$, we say gradients are $\underline{c}$-conflicted (with $\underline{c} \geq 0$) if, for all task pairs $j, k, j \neq k$,

$$-\underline{c}\big(\|\nabla \mathcal{L}_j(\boldsymbol{\theta})\|^2 + \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2\big) \leq \langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \rangle .$$

The above definition is implied by a lower bound on the gradients cosine similarity. In particular, if $\frac{\langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \rangle}{\|\mathcal{L}_j(\boldsymbol{\theta})\|\|\mathcal{L}_k(\boldsymbol{\theta})\|} \geq -\tilde{c}$, then the gradients are $\underline{c}$-conflicted for $\underline{c} = \tilde{c}/2$. Therefore, experiments in section 3.1 show that $\underline{c}$ is typically small for LLM training. The reader is also referred to Figures 5 and 6 in Appendix E.2, where we plot the ratio $\frac{\langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \rangle}{\|\mathcal{L}_j(\boldsymbol{\theta})\|^2 + \|\mathcal{L}_k(\boldsymbol{\theta})\|^2}$ for the same experiment in Figure 2.

While Definition 3.2 quantifies the conflict between gradients, we also observed in section 3.1 that the gradients of different tasks are also not completely aligned. To quantify the level of alignment, we define the following concept:

**Definition 3.3.** For a given point $\boldsymbol{\theta}$, we say that the gradients are $\bar{c}$-aligned (with $\bar{c} \geq 0$) if, for all task pairs $j, k, j \neq k$,

$$\langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \rangle \leq \bar{c} \|\nabla \mathcal{L}_j(\boldsymbol{\theta})\|_2 \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|_2.$$

While $\bar{c} = 1$ and $\underline{c} = 1/2$ always hold, smaller values allow for more refined analysis. Notably, when both $\bar{c}$ and $\underline{c}$ are small, the value of $\|\nabla \mathcal{L}(\boldsymbol{\theta})\|$ is small if and only if $\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|$ is small for all $k$ (see Lemma F.1 in Appendix F).

To proceed, we make the following standard assumptions.

**Assumption 3.4.** For all tasks $k \in \{1, \ldots, K\}$, the gradients are $L$-Lipschitz, unbiased, and have bounded variance:

$$\|\nabla \mathcal{L}_k(\boldsymbol{\theta}_1) - \nabla \mathcal{L}_k(\boldsymbol{\theta}_2)\| \leq L\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|, \quad \forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \tag{5}$$

$$\mathbb{E}_{x \sim \mathcal{D}_k}[\nabla \ell_k(\boldsymbol{\theta}; x)] = \nabla \mathcal{L}_k(\boldsymbol{\theta}), \quad \forall \boldsymbol{\theta} \tag{6}$$

$$\mathbb{E}_{x \sim \mathcal{D}_k}[\|\nabla \ell_k(\boldsymbol{\theta}; x) - \nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2] \leq \sigma_k^2, \quad \forall \boldsymbol{\theta} \tag{7}$$

Using a Mix batch with $b_k$ samples per task $k$, the estimated gradient follows equation (2). The next theorem characterizes the descent obtained under low conflict conditions:

**Theorem 3.5.** *Suppose Assumption 3.4 holds and the gradients are $\underline{c}$-conflicted and $\bar{c}$-aligned at $\boldsymbol{\theta}_t$ with $\underline{c} < \frac{1}{K-2+b/b_k}, \forall k$. Moreover, assume the gradient is computed according to the mix* *sampling equation (2). Then,*

$$\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_t - \eta \mathbf{g}_t)] \leq \mathcal{L}(\boldsymbol{\theta}_t) + \sum_{k=1}^{K} b_k \Big( -\frac{\eta}{b}\beta \|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t)\|^2 + \frac{L\eta^2}{2b^2}\sigma_k^2 \Big) + \sum_{k=1}^{K} b_k^2 \frac{L\eta^2}{2b^2}\gamma \|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t)\|^2 \tag{8}$$

*where* $\beta \triangleq \min_k (1 + \underline{c}(-K + 2 - \frac{b}{b_k}))$ *and* $\gamma \triangleq 1 + \bar{c}(K-1)$*.*

A formal proof is provided in Theorem F.3 (Appendix F). To maximize descent in Mix sampling, we minimize the right-hand side of equation (8). Assuming a large $b$, we relax $b_k$ to continuous values $w_k = b_k/b$ and solve:

$$\min_{w_1,\ldots,w_K \geq 0} \sum_{k=1}^{K} w_k \lambda_k + \frac{1}{2} w_k^2 \kappa_k \ \text{ s.t. } \sum_{k=1}^{K} w_k = 1 \tag{9}$$

where $\lambda_k \triangleq -\eta\beta\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \frac{L\eta^2}{2b}\sigma_k^2$ and $\kappa_k \triangleq L\eta^2\gamma\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2$. Using KKT conditions, the optimal solution is given by

$$w_k^* = \max\left\{0, -\frac{\mu + \lambda_k}{\kappa_k}\right\} \tag{10}$$

where $\mu$ is chosen such that $\sum_{k=1}^{K} w_k^* = 1$ (see Lemma F.2, Appendix F). This leads to the conceptual version of PiKE (Positive gradient Interactions-based K-task weight Estimator), summarized in Algorithm 2 in Appendix B.

The conceptual version of PiKE (Algorithm 2) adaptively adjusts sampling weights. This adaptive adjustment makes the stochastic gradients biased, i.e., $\mathbb{E}[\mathbf{g}_t] \neq \nabla \mathcal{L}(\boldsymbol{\theta}_t)$. Due to this introduced bias, the classical convergence results of SGD can no longer be applied. The following theorem establishes the convergence of conceptual PiKE:

**Theorem 3.6.** *Suppose the assumptions in Theorem 3.5 hold and the Conceptual PiKE Algorithm (Algorithm 2) initialized at* $\boldsymbol{\theta}_0$ *with the SGD optimizer in Step 10 of the algorithm. Let* $\Delta_L = \mathcal{L}(\boldsymbol{\theta}_0) - \min_{\boldsymbol{\theta}} \mathcal{L}(\theta)$ *and* $\sigma_{\max} = \max_k \sigma_k$*. Suppose* $\delta > 0$ *is a given constant and the stepsize* $\eta \leq \frac{\beta\delta}{L\sigma_{\max}^2/b + L\eta\delta}$*. Then, after* $T = \frac{2\beta\Delta_L}{\eta\delta}$ *iterations, Algorithm 2 finds a point* $\bar{\boldsymbol{\theta}}$ *such that*

$$\mathbb{E}\|\nabla \mathcal{L}_k(\bar{\theta})\|^2 \leq \delta, \quad \forall k = 1, \ldots, K. \tag{11}$$

*Moreover, if we choose* $\eta = \frac{\beta\delta}{L\sigma_{\max}^2/b + L\eta\delta}$*, then the Conceptual PiKE algorithm requires at most*

$$\bar{T} = \frac{2L\Delta_L(\sigma_{\max}^2/b + \gamma\delta)}{\delta^2\beta^2}$$

*iterations to find a point satisfying equation (11).*

The proof of this theorem is provided in Theorem F.4 in Appendix F. This theorem states that with enough steps, the gradient of all task losses become small. It is also worth noting that the gradient norm becomes small with the iteration complexity $T = O(1/\delta^2)$, which is the best known rate for nonconvex smooth stochastic setting.

### 3.3 PiKE: Simplified Computationally Efficient Version

Solving equation (9) requires estimating $\{\sigma_k\}_{k=1}^K$ and $\{\|\nabla \mathcal{L}_k(\theta_t)\|^2\}_{k=1}^K$, which necessitates large batch computations, slowing convergence. To speed up the algorithm, we update these estimates every $T$ iterations. However, this can cause abrupt changes in sampling weights $(w_1, \ldots, w_K)$, leading to instability, especially with optimizers like Adam, where sudden shifts may disrupt momentum estimates. To mitigate this, we update $(w_1, \ldots, w_K)$ using a single mirror descent step on equation (9), ensuring gradual adjustments:

$$w_k \leftarrow w_k \exp\left(\alpha\eta(\beta - L\eta\gamma w_k)\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 - \frac{\alpha L\eta^2}{2b}\sigma_k^2\right)$$

followed by normalization: $\mathbf{w} \leftarrow \mathbf{w}/\|\mathbf{w}\|_1$, where $\alpha$ is the mirror descent step size.

**Algorithm 1** PiKE: Positive gradient Interaction-based K-task weights Estimator

1: **Input:** $\boldsymbol{\theta}$, $T$, total batch size $b$, task $k$ dataset $\mathcal{D}_k$, hyperparameters $\zeta_1$ and $\zeta_2$, prior weights $\mathbf{w}'$
2: **Initialize:** $w_k \leftarrow 1/K$ or $w_k \leftarrow w'_k$
3: **for** $t = 0, 1, \ldots$ **do**
4: **if** $t \mod T = 0$ **then**
5: Estimate $\|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t)\|^2$ and $\sigma_k^2$ for every $k$
6: $w_k \leftarrow w_k \exp\left(\zeta_1 \|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t)\|^2 - \frac{\zeta_2}{2b}\sigma_k^2\right)$
7: $\mathbf{w} \leftarrow \mathbf{w}/\|\mathbf{w}\|_1$
8: $(b_1, \ldots, b_K) \leftarrow \text{round}(b(w_1, \ldots, w_K))$
9: **end if**
10: Sample $b_k$ data points from each task $k$
11: Compute the gradient $\mathbf{g}$ using the estimates samples
12: Update: $\boldsymbol{\theta}_{t+1} \leftarrow \text{Optimizer}(\eta, \boldsymbol{\theta}_t, \mathbf{g})$
13: **end for**

Fine-tuning $L, \gamma, \alpha$, and $\beta$ can be challenging, but we simplify this by noting two observations: 1) The coefficient of $\sigma_k^2$ is constant, independent of $w_k$. 2) For small $\eta$ and $w_k < 1$, the coefficient of $\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|$ remains nearly constant: $\alpha\eta(\beta - L\eta\gamma w_k) \approx \alpha\eta\beta$. Thus, in practice, we use tunable constant coefficients for variance and gradient norm terms, simplifying implementation. The final algorithm is summarized in Algorithm 1.

### 3.4 PiKE: Fairness Considerations Across Tasks

Algorithm 1 is designed to minimize the average loss across tasks as in equation (1). To ensure fair learning across all tasks, we can consider a fairness-promoting objective based on *tilted empirical risk minimization* (Li et al., 2020), also known as the *$\alpha$-fairness utility* (Mo & Walrand, 2000):

$$\min_{\boldsymbol{\theta}} \widetilde{\mathcal{L}}(\tau; \boldsymbol{\theta}) := \frac{1}{\tau} \log \left( \sum_{k=1}^{K} e^{\tau \mathcal{L}_k(\boldsymbol{\theta})} \right). \tag{12}$$

This formulation reduces to equation (1) as $\tau \to 0$, while for $\tau > 0$, it promotes fairness. In the limit $\tau \to \infty$, it optimizes for the worst-case task loss, i.e., $\max_k \mathcal{L}_k(\boldsymbol{\theta})$, ensuring no task is disproportionately neglected. Moderate values of $\tau$ balance fairness and performance.

We can use Fenchel duality (Rockafellar, 2015), to connect the objective in equation (12) to a weighted version of equation (1) through the following lemma:

**Lemma 3.7.** *Let $\mathbf{x} \in \mathbb{R}_+^K$ and $\tau > 0$. Then,*

$$\log\left(\sum_{k=1}^{K} e^{\tau x_k}\right) = \max_{\substack{\mathbf{y} \in \mathbb{R}_+^K \\ \sum_{k=1}^K y_k = \tau}} \left( \sum_{k=1}^{K} y_k x_k - \sum_{k=1}^{K} \frac{y_k}{\tau} \log\left(\frac{y_k}{\tau}\right) \right)$$

The proof of this Lemma F.5 can be found in Appendix F.3. Using this lemma, equation (12) can be rewritten as

$$\min_{\boldsymbol{\theta}} \quad \max_{\substack{\mathbf{y} \in \mathbb{R}_+^K \\ \sum_{k=1}^K y_k = \tau}} \sum_{k=1}^{K} y_k \mathcal{L}_k(\boldsymbol{\theta}) - \sum_{k=1}^{K} \frac{y_k}{\tau} \log\left(\frac{y_k}{\tau}\right),$$

ehrtr the optimal $\mathbf{y}$, for a fixed $\boldsymbol{\theta}$, has a closed-form solution:

$$y_k^\star = \frac{\tau e^{\tau \mathcal{L}_k(\boldsymbol{\theta}) - 1}}{\sum_{j=1}^K e^{\tau \mathcal{L}_j(\boldsymbol{\theta}) - 1}}, \quad \forall k.$$

(see Lemma F.6 in Appendix F.3). On the other hand, fixing $\mathbf{y}$, the problem reduces to a weighted minimization over tasks, where *regular PiKE sampling* with proper weights $y_k$ in front of each loss can be applied to determine the optimal mixing strategy. This leads to *fair-PiKE* algorithm, described in Appendix C, which balances overall loss minimization and fair learning of all tasks.

Table 1: We report the perplexities (lower the better) on the validation split of multilingual C4 datasets. We also compare the accuracies (%, higher the better) of different models on $\overline{\text{HellaSwag}}$ and its corresponding translated version. **Bolding** indicates the best model in the task; $\overline{\text{Metrics}}$ means the average across different tasks. Additional results can be found in Table 8.

| | | C4 (en) | C4 (hi) | C4 (de) | | HellaSwag (en) | HellaSwag (hi) | HellaSwag (de) |
|---|---|---|---|---|---|---|---|---|
| | $\overline{\text{Perplexity}}$ ↓ | Perplexity ↓ | Perplexity ↓ | Perplexity ↓ | $\overline{\text{Accuracy}}$(%) ↑ | 0-shot ↑ | 0-shot ↑ | 0-shot ↑ |
| **C4 (en), C4 (hi), and C4 (de) datasets, GPT-2 large style, 1B params, 36 Layers default, 120K training steps** | | | | | | | | |
| Mix | 8.29 | 11.13 | 4.45 | 9.29 | 27.5 | 28.1 | 27.1 | 27.6 |
| Round-Robin | 8.41 | 11.31 | 4.97 | 9.46 | 26.5 | 27.6 | 26.7 | 26.3 |
| Random | 8.48 | 11.38 | 4.54 | 9.55 | 26.6 | 27.0 | 26.9 | 26.1 |
| PiKE | 9.56 | **9.49** | 5.32 | 13.87 | 28.7 | **33.0** | 27.2 | 26.2 |
| Fair-PiKE ($\tau = 1$) | 8.29 | 11.12 | **4.46** | 9.31 | 27.9 | 28.3 | **27.4** | 28.0 |
| Fair-PiKE ($\tau = 3$) | **8.18** | 10.14 | 4.93 | 9.49 | **28.9** | 31.3 | 27.3 | 28.1 |
| Fair-PiKE ($\tau = 5$) | 8.42 | 10.02 | 6.30 | **8.94** | **28.9** | 31.2 | 26.9 | **28.6** |

Table 2: We report perplexity (lower is better) on the validation split of the GLaM datasets, averaging perplexities across six domains when applicable or reporting a single perplexity when only training with a single domain. We also compare the accuracies (%, higher the better) of different models on four different Q/A tasks. HellaSwag and ArcE tasks have 4 choices, CSQA has 5 choices, and PIQA has 2 choices. PiKE (Uniform) means PiKE using initial sampling weights of $1/6$ for each task and PiKE (GLaM) means PiKE using GLaM tuned weights as initial task weights. **Bolding** indicates the best model in the task, $\overline{\text{Metrics}}$ means the average across different tasks, underlining indicates PiKE beating Mix, Round-Robin, Random methods. Additional results can be found in Table 9.

| | GLaM | | ArcE | CSQA | HellaSwag | PIQA |
|---|---|---|---|---|---|---|
| | $\overline{\text{Perplexity}}$ ↓ | $\overline{\text{Accuracy}}$(%) ↑ | 7-shot ↑ | 7-shot ↑ | 7-shot ↑ | 7-shot ↑ |
| **Six domains of GLaM dataset, GPT-2 large style, 750M params, 36 layers default** | | | | | | |
| Mix | **12.77** | 46.4 | 47.2 | 39.6 | 37.9 | 60.9 |
| Round-Robin | 12.98 | 44.3 | 43.5 | 36.7 | 36.8 | 60.3 |
| Random | 12.99 | 42.7 | 41.7 | 34.2 | 36.6 | 58.2 |
| GLaM | 13.20 | 45.3 | 46.9 | 39.8 | **38.0** | 56.4 |
| DoReMi | 13.25 | 46.5 | 48.6 | 40.1 | 37.5 | 59.6 |
| PiKE (Uniform) | 13.22 | 47.6 | 49.6 | 43.2 | 37.2 | 60.4 |
| PiKE (GLaM) | 13.35 | **48.1** | **49.8** | **43.5** | **38.0** | **61.2** |

# 4 Experiments

We evaluate PiKE in two multitask pretraining scenarios: 1) *Pretraining language models on multilingual mC4 dataset* (Xue, 2020), a dataset covering diverse languages from Common Crawl corpus. 2) *Pretraining language models on the GLaM dataset* (Du et al., 2022), an English dataset spanning six domains. As we will see, across multiple model sizes (110M, 270M, 750M, and 1B parameters), *PiKE consistently outperforms static and heuristic data mixing methods*. For 1B models trained on multilingual C4 (en, hi), PiKE improves average downstream accuracy by **7.1%** and reaches baseline accuracy **1.9× faster**. For 750M models pre-trained on the GLaM dataset, PiKE improves averge downstream accuracy by **3.4%** over DoReMi (Xie et al., 2024) and **6.2%** over GLaM's original strategy.

## 4.1 Experiment Setup

**Baselines:** For multilingual pretraining, we compare five sampling strategies: 1. *Mix*, 2. *Round-Robin*, 3. *Random*, 4. *PiKE*, and 5. *fair-PiKE*. For GLaM-based pretraining, we evaluate: 1. *Mix*, 2. *GLaM* (Du et al., 2022), 3. *DoReMi* (Xie et al., 2024), and 4. *PiKE*. DoReMi trains a small proxy model for weight estimation, while GLaM assigns static domain weights based on downstream performance of smaller models. In contrast, PiKE dynamically adjusts weights during training based on gradient information. Hence, PiKE does not require another smaller model and is computationally much more efficient than DoReMi and GLaM.

**Datasets:** For multilingual experiments, we use mC4 (Xue, 2020), focusing on English (en), Hindi (hi), and German (de). An overview of these datasets is provided in Table 3. For GLaM-based experiments, we use the six-domain GLaM dataset (Du et al., 2022), with domain weights from (Du et al., 2022; Xie et al., 2024). Details regarding the GLaM dataset and the domain weights used by GLaM and DoReMi are presented in Table 4.

**Evaluation:** Perplexity is measured on held-out validation data. Downstream evaluation follows the OLMES suite (Gu et al., 2024). For multilingual downstream tasks, we use multilingual HellaSwag (Dac Lai et al., 2023), covering 26 languages. For models trained on GLaM, we evaluate on downstream tasks ARC-Easy (Clark et al., 2018), CommonsenseQA (Talmor et al., 2018), PIQA (Bisk et al., 2019), and HellaSwag (Zellers et al., 2019).

Further details on our experimental setup and evaluation are in Appendix D.

### 4.2 PiKE Outperforms Mix, Round-Robin, and Random in Multilingual Pretraining

Table 1 presents results for pretraining a 1B multilingual GPT-2 model (Radford et al., 2019) on English, Hindi, and German, with additional results in Table 8. We evaluate GPT-2 models at two scales (270M and 1B parameters) across two language settings: (1) English and Hindi, and (2) English, Hindi, and German.

We observe that *PiKE and its fair variation consistently achieve the highest average accuracy of downstream tasks* across all language settings and model scales, demonstrating its effectiveness in multilingual pretraining.

We also observe that *fair-PiKE balances fairness among tasks.* We pre-trained 1B models using Fair-PiKE with different fairness parameters $\tau \in \{1, 3, 5\}$. Higher $\tau$ values promotes greater fairness by reducing the gap between task losses. At $\tau = 5$, perplexity values across tasks become more uniform, indicating improved fairness. Notably, Fair-PiKE with $\tau = 3$ achieves the best balance, yielding the lowest perplexity and highest downstream performance. These results highlight the benefits of incorporating fairness considerations in pretraining.

### 4.3 PiKE Outperforms DoReMi, GLaM, and Static Mix in Pretraining with GLaM Datasets

Table 2 presents results for pretraining a 750M multilingual GPT-2 model on the GLaM dataset, with additional results in Table 9. We evaluate two model sizes (110M and 750M) across six domains.

*PiKE consistently achieves the highest average performance.* In both 110M and 750M configurations, PiKE outperforms DoReMi, GLaM, and Mix in downstream accuracy. For 750M models, PiKE improves the average downstream task accuracy by **3.4%** over DoReMi and **6.2%** over GLaM. For 110M models, PiKE achieves **37.8%** accuracy, surpassing DoReMi (**36.0%**) and GLaM (**35.3%**). Unlike DoReMi and GLaM, PiKE achieves these improvements without additional computational overhead, as DoReMi requires training a proxy model and GLaM involves tuning weights based on smaller models.

*PiKE benefits from apriori downstream-tuned weights.* We evaluate PiKE with two initializations: (1) uniform weights $b_k = b/K$ and (2) GLaM-tuned weights. In both small and large GPT-2 configurations, PiKE benefits from utilizing already fine tuned weights as initialization, achieving **48.1%** accuracy with GLaM-tuned weights vs. **47.6%** with uniform initialization. This shows that PiKE can effectively leverage pre-existing fine-tuned weights while still outperforming other methods with uniform initialization.

*Mixing datasets improves language model generalization.* We compare models trained on individual domains to those trained on mixed-domain datasets. Table 9 shows that single-domain training underperforms compared to mixed-domain training, even with simple Mix sampling. This reinforces the importance of diverse data for pretraining and aligns with prior work (Liu et al., 2024c; Hoffmann et al., 2022).

*Discussion on perplexity.* Table 9 reveals that validation perplexity does not always align with downstream performance. For instance, while Mix sampling yields lower perplexity in 750M models, PiKE achieves better downstream accuracy. This aligns with prior findings (Tay et al., 2021; Liu et al., 2023; Wettig et al., 2024), suggesting that perplexity alone is not a reliable performance metric.

## CONCLUSION

In this work, we introduced PiKE, an adaptive data mixing algorithm for multitask learning that dynamically adjusts task sampling based on gradient interactions. Unlike prior approaches that focus on mitigating gradient conflicts, PiKE leverages the positive gradient interactions commonly observed in large-scale language model training. Our theoretical analysis established the convergence guarantees of PiKE, while empirical results demonstrated its effectiveness across diverse pretraining scenarios. Furthermore, we extended PiKE to incorporate fairness considerations, ensuring balanced learning across tasks. Our results indicate that Fair-PiKE effectively reduces task performance disparities while maintaining strong overall model performance.

A key limitation of our work is that PiKE does not explicitly account for data abundance when adjusting sampling weights. Future work could explore integrating dataset prevalence into the adaptive mixing strategy to further optimize learning efficiency. Additionally, extending PiKE to other domains beyond language modeling presents an exciting direction for future research.

## REFERENCES

Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S Morcos. Semdedup: Data-efficient learning at web-scale through semantic deduplication. *arXiv preprint arXiv:2303.09540*, 2023.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

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

Y Bisk, R Zellers, R Le Bras, J Gao, and Y Choi. Reasoning about physical commonsense in natural language, 2019.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, pp. 794–803. PMLR, 2018.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Viet Dac Lai, Chien Van Nguyen, Nghia Trung Ngo, Thuat Nguyen, Franck Dernoncourt, Ryan A Rossi, and Thien Huu Nguyen. Okapi: Instruction-tuned large language models in multiple languages with reinforcement learning from human feedback. *arXiv e-prints*, pp. arXiv–2307, 2023.

Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3150–3158, 2016.

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. *arXiv preprint arXiv:2302.05442*, 2023.

Jean-Antoine Désidéri. Multiple-gradient descent algorithm (mgda) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5-6):313–318, 2012.

Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. Glam: Efficient scaling of language models with mixture-of-experts. In *International Conference on Machine Learning*, pp. 5547–5569. PMLR, 2022.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Colin Gaffney, Dinghua Li, Ruoxin Sang, Ayush Jain, and Haitang Hu. Orbax, 2023. URL `http://github.com/google/orbax`.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Ce Ge, Zhijian Ma, Daoyuan Chen, Yaliang Li, and Bolin Ding. Data mixing made efficient: A bivariate scaling law for language model pretraining. *arXiv preprint arXiv:2405.14908*, 2024.

Google. Grain - feeding jax models, 2023. URL `http://github.com/google/grain`.

Yuling Gu, Oyvind Tafjord, Bailey Kuehl, Dany Haddad, Jesse Dodge, and Hannaneh Hajishirzi. Olmes: A standard for language model evaluations. *arXiv preprint arXiv:2406.08446*, 2024.

Kelvin Guu, Albert Webson, Ellie Pavlick, Lucas Dixon, Ian Tenney, and Tolga Bolukbasi. Simfluence: Modeling the influence of individual training examples by simulating training runs. *arXiv preprint arXiv:2303.08114*, 2023.

Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2023. URL `http://github.com/google/flax`.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. An empirical analysis of compute-optimal large language model training. *Advances in Neural Information Processing Systems*, 35:30016–30030, 2022.

Yiding Jiang, Allan Zhou, Zhili Feng, Sadhika Malladi, and J Zico Kolter. Adaptive data optimization: Dynamic sample selection with scaling laws. *arXiv preprint arXiv:2410.11820*, 2024.

Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*, pp. 1–12, 2017.

Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, et al. The bigscience roots corpus: A 1.6 tb composite multilingual dataset. *Advances in Neural Information Processing Systems*, 35:31809–31826, 2022.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. *arXiv preprint arXiv:2107.06499*, 2021.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. Measuring the intrinsic dimension of objective landscapes. In *International Conference on Learning Representations (ICLR)*, 2018. https://arxiv.org/abs/1804.08838.

Tian Li, Ahmad Beirami, Maziar Sanjabi, and Virginia Smith. Tilted empirical risk minimization. *arXiv preprint arXiv:2007.01162*, 2020.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024a.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890, 2021.

Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. Famo: Fast adaptive multitask optimization. *Advances in Neural Information Processing Systems*, 36, 2024b.

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*, pp. 22188–22214. PMLR, 2023.

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. *arXiv preprint arXiv:2407.01492*, 2024c.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Association for Computational Linguistics*, 2015.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019. https://openreview.net/forum?id=Bkg6RiCqY7.

Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*, 2015.

Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3994–4003, 2016.

Jeonghoon Mo and Jean Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on networking*, 8(5):556–567, 2000.

Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. Multi-task learning as a bargaining game. *arXiv preprint arXiv:2202.01017*, 2022.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Hamza Alobeidli, Alessandro Cappelli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data only. *Advances in Neural Information Processing Systems*, 36:79155–79172, 2023.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners, 2019. https://openai.com/blog/better-language-models/.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Jie Ren, Samyam Rajbhandari, Reza Yazdani Aminabadi, Olatunji Ruwase, Shuangyan Yang, Minjia Zhang, Dong Li, and Yuxiong He. {ZeRO-Offload}: Democratizing {Billion-Scale} model training. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pp. 551–564, 2021.

Ralph Tyrell Rockafellar. Convex analysis:(pms-28). *Princeton university press*, 2015.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. Latent multi-task architecture learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 4822–4829, 2019.

Noveen Sachdeva, Benjamin Coleman, Wang-Cheng Kang, Jianmo Ni, Lichan Hong, Ed H Chi, James Caverlee, Julian McAuley, and Derek Zhiyuan Cheng. How to train data-efficient llms. *arXiv preprint arXiv:2402.09668*, 2024.

Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, et al. Dolma: An open corpus of three trillion tokens for language model pretraining research. *arXiv preprint arXiv:2402.00159*, 2024.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*, 2018.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. Scale efficiently: Insights from pre-training and fine-tuning transformers. *arXiv preprint arXiv:2109.10686*, 2021.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024a.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024b.

Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3614–3633, 2021.

Huy V Vo, Vasil Khalidov, Timothée Darcet, Théo Moutakanni, Nikita Smetanin, Marc Szafraniec, Hugo Touvron, Camille Couprie, Maxime Oquab, Armand Joulin, et al. Automatic data curation for self-supervised learning: A clustering-based approach. *arXiv preprint arXiv:2405.15613*, 2024.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32, 2019.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. *arXiv preprint arXiv:2010.05874*, 2020.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. Qurating: Selecting high-quality data for training language models. *arXiv preprint arXiv:2402.09739*, 2024.

Mitchell Wortsman, Peter J Liu, Lechao Xiao, Katie Everett, Alex Alemi, Ben Adlam, John D Co-Reyes, Izzeddin Gur, Abhishek Kumar, Roman Novak, et al. Small-scale proxies for large-scale transformer training instabilities. *arXiv preprint arXiv:2309.14322*, 2023.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared llama: Accelerating language model pre-training via structured pruning. *arXiv preprint arXiv:2310.06694*, 2023.

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. Doremi: Optimizing data mixtures speeds up language model pretraining. *Advances in Neural Information Processing Systems*, 36, 2024.

Derrick Xin, Behrooz Ghorbani, Justin Gilmer, Ankush Garg, and Orhan Firat. Do current multi-task optimization methods in deep learning even help? *Advances in neural information processing systems*, 35:13597–13609, 2022.

L Xue. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*, 2020.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. Byt5: Towards a token-free future with pre-trained byte-to-byte models. corr, abs/2105.13626. *arXiv preprint arXiv:2105.13626*, 2021.

Enneng Yang, Junwei Pan, Ximei Wang, Haibin Yu, Li Shen, Xihua Chen, Lei Xiao, Jie Jiang, and Guibing Guo. Adatask: A task-aware adaptive learning rate approach to multi-task learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 10745–10753, 2023.

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

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33: 5824–5836, 2020.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Yushun Zhang, Congliang Chen, Tian Ding, Ziniu Li, Ruoyu Sun, and Zhi-Quan Luo. Why transformers need adam: A hessian perspective. *arXiv preprint arXiv:2402.16788*, 2024.

# A Related Work

**Data Curation and Selection.** The effectiveness of language models heavily depends on the quality of the pre-training corpus. Consequently, significant efforts have been made to enhance pre-training data. These efforts include heuristic-based filtering (Raffel et al., 2020; Rae et al., 2021; Laurençon et al., 2022; Penedo et al., 2023; Soldaini et al., 2024) and deduplication (Abbas et al., 2023; Lee et al., 2021; Chowdhery et al., 2022; Dubey et al., 2024). Recently, Vo et al. (2024) proposed an automated method for constructing large, diverse, and balanced datasets for self-supervised learning by applying hierarchical k-means clustering. Sachdeva et al. (2024) introduced techniques that leverage instruction-tuned models to assess and select high-quality training examples, along with density sampling to ensure diverse data coverage by modeling the data distribution. Additionally, Guu et al. (2023) simulated training runs to model the non-additive effects of individual training examples, enabling the analysis of their influence on a model's predictions.

**Multitask Learning Optimization** The approach most closely related to our method is multitask learning (MTL) optimization, which modifies gradient updates to mitigate gradient conflicts—situations where task gradients point in opposing directions, slowing down optimization (Vandenhende et al., 2021; Yu et al., 2020). The Multiple Gradient Descent Algorithm (MGDA) (Désidéri, 2012; Sener & Koltun,

2018) updates the model by optimizing the worst improvement across all tasks, aiming for equal descent in task losses. Projected Gradient Descent (PCGrad) (Yu et al., 2020) modifies task gradients by iteratively removing conflicting components in a randomized order, ensuring that updates do not interfere destructively across tasks. Conflict-Averse Gradient Descent (CAGRAD) (Liu et al., 2021) optimizes for the worst task improvement while ensuring a decrease in the average loss. NASHMTL (Navon et al., 2022) determines gradient directions by solving a bargaining game that maximizes the sum of log utility functions. While these methods improve performance, they introduce significant computational and memory overhead, making them impractical for large-scale models with numerous tasks (Xin et al., 2022). Similar challenges exist in AdaTask (Yang et al., 2023), which improves multitask learning by balancing parameter updates using task-wise adaptive learning rates, mitigating task dominance, and enhancing overall performance. Unlike previous approches that requires requiring $O(K)$ storage for task gradients (e.g. PCGrad) or optimizer states (e.g. AdaTask), FAMO (Liu et al., 2024b) balances task loss reductions efficiently using $O(1)$ space and time. However, these methods fail to exploit the *non-conflicting* interactions among tasks, focusing instead on resolving conflicts that seldom arise. This highlights the need for a new approach that actively leverages lack of gradient conflicts to enhance training efficiency.

Another line of work focuses on adjusting the domain mixture to improve data efficiency during training (Xie et al., 2024; Xia et al., 2023; Jiang et al., 2024). However, these methods require a target loss for optimization, which has been shown to not always correlate with downstream performance (Tay et al., 2021; Liu et al., 2023; Wettig et al., 2024). In contrast, our method leverages the absence of gradient conflict and the presence of positive gradient interactions between tasks or domains. This approach provides a more reliable and effective way to enhance the final model's performance.

# B PiKE: Conceptual Version

Here, we present the conceptual (basic) version of PiKE. As discussed in the main text, this approach lacks computational efficiency due to the frequent estimation of the norm and the variance of the per-task gradient.

**Algorithm 2** Conceptual version of PiKE: Positive gradient Interaction-based K-task weights Estimator

1: **Input:** $\boldsymbol{\theta}$, total batch size $b$, stepsize $\eta$, task $k$ dataset $\mathcal{D}_k$, constants $\beta$, $L$, $\gamma$, and prior weights $\mathbf{w}'$

2: **Initialize:** $w_k \leftarrow 1/K$ or $w_k \leftarrow w'_k, \forall k$
3: **for** $t = 0, 1, \dots$ **do**
4: Estimate $\|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t)\|^2$ and $\sigma_k^2$ for every $k$
5: Compute $\lambda_k \triangleq -\eta\beta\|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t)\|^2 + \frac{L\eta^2}{2b}\sigma_k^2$ and $\kappa_k \triangleq L\eta^2\gamma\|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t)\|^2$
6: set $w_k^* = \max\{0, -\frac{\mu+\lambda_k}{\kappa_k}\}$ where $\mu$ is found (by bisection) such that $\sum_{k=1}^K w_k^* = 1$
7: Set $(b_1, \dots, b_K) \leftarrow \text{round}(b(w_1^*, \dots, w_K^*))$
8: Sample $b_k$ data points from each task $k$
9: Compute the gradient $\mathbf{g}$ using the estimates samples
10: Update: $\boldsymbol{\theta}_{t+1} \leftarrow \text{Optimizer}(\eta, \boldsymbol{\theta}_t, \mathbf{g})$
11: **end for**

As discussed in section 3.3, this algorithm is computationally inefficient as it requires estimating $\nabla \mathcal{L}_k(\theta_t)$ and $\sigma_k$ at each iteration. To improve efficiency, we introduced modifications that led to the development of the PiKE algorithm (Algorithm 1 in the main body).

# C Fair-PiKE: Fairness Considerations Across Tasks

Here, we present the *fair-PiKE* algorithm in more detail. As discussed in the main body, the main difference with PiKE is that the fair version requires the computation of the coefficients

$$y_k^\star = \frac{\tau e^{\tau \mathcal{L}_k(\boldsymbol{\theta})-1}}{\sum_{k=1}^K e^{\tau \mathcal{L}_k(\boldsymbol{\theta})-1}}, \forall k$$

Then updating the sampling weights by

$$w_k \leftarrow w_k \exp\left((y_k^\star)^2 \zeta_1 \|\nabla \mathcal{L}_k(\mathbf{w})\|^2 - (y_k^\star)^2 \frac{\zeta_2}{2b} \sigma_k^2\right), \quad \forall k$$

The overall algorithm is summarized in Algorithm 3. For our experiments, we evaluate three different values of $\tau$: 1, 3, and 5. A larger $\tau$ results in a stronger balancing effect between different tasks.

**Algorithm 3** *fair-PiKE*: Fairness considerations across tasks

1: **Input:** $\boldsymbol{\theta}$, $T$, total batch size $b$, task $k$ dataset $\mathcal{D}_k$, hyperparameters $\zeta_1$ $\zeta_2, \tau$, prior weights $\mathbf{w}'$
2: **Initialize:** $w_k \leftarrow 1/K$ or $w_k \leftarrow w'_k$
3: **for** $t = 0, 1, \ldots$ **do**
4:     **if** $t \mod T = 0$ **then**
5:         Estimate $\|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t)\|^2$, $\sigma_k^2$, and $\mathcal{L}_k(\boldsymbol{\theta}_t)$ for every $k$
6:         $y_k^\star = \frac{\tau e^{\tau \mathcal{L}_k(\boldsymbol{\theta})-1}}{\sum_{k=1}^K e^{\tau \mathcal{L}_k(\boldsymbol{\theta})-1}}$
7:         $w_k \leftarrow w_k \exp\left((y_k^\star)^2 \zeta_1 \|\nabla \mathcal{L}_k(\mathbf{w})\|^2 - (y_k^\star)^2 \frac{\zeta_2}{2b} \sigma_k^2\right)$
8:         $\mathbf{w} \leftarrow \mathbf{w}/\|\mathbf{w}\|_1$
9:         $(b_1, \ldots, b_K) \leftarrow \text{round}(b(w_1, \ldots, w_K))$
10:     **end if**
11:     Sample $b_k$ data points from each task $k$
12:     Compute the gradient $\mathbf{g}$ using the estimates samples
13:     Update: $\boldsymbol{\theta}_{t+1} \leftarrow \text{Optimizer}(\eta, \boldsymbol{\theta}_t, \mathbf{g})$
14: **end for**

# D Experiments Setup

## D.1 Dataset Details

Our experiments construct two primary scenarios for multitask learning: multilingual tasks and diverse task mixtures spanning multiple domains. We consider two widely-used datasets for our study: mC4 (Xue, 2020) and GLaM (Du et al., 2022).

**mC4 Dataset** The mC4 dataset (Xue, 2020) is a multilingual text corpus derived from the Common Crawl web archive, covering a diverse range of languages. It has been widely used for pretraining multilingual models, such as mT5 (Xue, 2020) and ByT5 (Xue et al., 2021). The dataset is curated by applying language-specific filtering to extract high-quality text, ensuring a balanced representation across languages. Mixture weights for training models on mC4 are often chosen based on token counts. In our cases, we mainly focus on English (en), Hindi (hi), and German (de). We report their details in Table 3.

Table 3: Partial statistics of the mC4 corpus, totaling 6.3T tokens.

| ISO code | Language | Tokens (B) |
|---|---|---|
| en | English | 2,733 |
| hi | Hindi | 24 |
| de | German | 347 |

**GLaM Dataset** The GLaM dataset (Du et al., 2022) comprises English text from six distinct sources and has been used to train the GLaM series models and PaLM (Chowdhery et al., 2023). Mixture weights for GLaM training were determined based on small model performance (Du et al., 2022), while (Xie et al., 2024) employed group distributionally robust optimization (Group DRO) to compute domain-specific weights. Table 4 summarizes the six domains in the GLaM dataset and the mixture weights selected by GLaM and DoReMi. We use these weights as oracle baselines for comparison with PiKE, which dynamically adjusts task weights over time using gradient information, unlike the fixed weights employed by GLaM and DoReMi.

Table 4: GLaM dataset (Du et al., 2022) and fixed mixture weights used in GLaM (Du et al., 2022) and DoReMi (Xie et al., 2024).

| Dataset | Tokens (B) | Weight chosen by GLaM (Du et al., 2022) | Weight chosen by DoReMi (Xie et al., 2024) |
|---|---|---|---|
| Filtered Webpages | 143 | 0.42 | 0.51 |
| Wikipedia | 3 | 0.06 | 0.05 |
| Conversations | 174 | 0.28 | 0.22 |
| Forums | 247 | 0.02 | 0.04 |
| Books | 390 | 0.20 | 0.20 |
| News | 650 | 0.02 | 0.02 |

Table 5: Architecture hyperparameters for different model scales used in the paper. All models are GPT-2-like decoder-only architectures. The multilingual models employ a vocabulary size of 250K, whereas GLaM training uses a vocabulary size of 32K. Differences in the total number of parameters arise due to the variation in vocabulary sizes.

| Size | # Params | Layers | Attention heads | Attention head dim | Hidden dim |
|---|---|---|---|---|---|
| GPT-2 small | 110M/270M | 12 | 12 | 64 | 768 |
| GPT-2 large | 750M/1B | 36 | 20 | 64 | 1280 |

### D.2 Training Details

Our experiments explore two distinct scenarios for multitask learning: multilingual training and diverse task mixtures spanning multiple domains. To achieve optimal results, we customize the training setups for each scenario and present them separately in this section. All training is performed from scratch.

**Multilingual Training** To address the complexities of tokenizing multilingual data, we utilize the mT5 tokenizer (Xue, 2020), which features a vocabulary size of 250K. Both GPT-2 small and GPT-2 large models are trained with a context length of 1024 and a batch size of 256. The AdamW optimizer (Loshchilov & Hutter, 2019) is employed with consistent hyperparameters and a learning rate scheduler. Additional details on hyperparameter configurations are provided in Appendix D.5.

**GLaM Training** For GLaM training, we use the T5 tokenizer (Raffel et al., 2020), implemented as a SentencePiece tokenizer trained on the C4 dataset with a vocabulary size of 32,000. Both GPT-2 small and GPT-2 large models are trained with a context length of 1024 and a batch size of 256. The AdamW optimizer (Loshchilov & Hutter, 2019) is used, and additional details on hyperparameters is in Appendix D.5.

### D.3 Model Architecture

The detailed architecture is summarized in Table 5. Our implementation utilizes pre-normalization (Radford et al., 2019) Transformers with qk-layernorm (Dehghani et al., 2023). Consistent with Chowdhery et al. (2022), we omit biases, and the layernorm (Ba et al., 2016) value remains set to the Flax (Heek et al., 2023) default of 1e-6. Additionally, we incorporate rotary positional embeddings (Su et al., 2021).

### D.4 Experimental Resource

All experiments are conducted on 8 Google TPUv4. The training time for GPT-2 small and GPT-2 large models for 120K steps are approximately 1 day and 2 days per run, respectively.

### D.5 Hyper-parameters

Table 6 shows the detailed hyperparameters that we used in all our experiments. We also report our hyperparameters grid for tuning PiKE in Table 7.

Table 6: Hyperparameter settings for our experiments.

| **Hyperparameters** | **Values** |
|---|---|
| Optimizer | AdamW ($\beta_1 = 0.95$, $\beta_2 = 0.98$) |
| Initial and final learning rate | $7e-6$ |
| Peak learning rate | $7e-4$ |
| Weight decay | 0.1 |
| Batch size | 256 |
| Context length | 1024 |
| Gradient clipping norm | 1.0 |
| Training step | $120,000$ |
| Warm-up step | $10,000$ |
| Schedule | Linear decay to final learning rate |

Table 7: Hyperparameter settings for running PiKE (Algorithm 1).

| **Hyperparameters** | **Values** |
|---|---|
| PiKE hyperparameter $\zeta_1$ | $\{0.025, 0.01, 0.75\}$ |
| PiKE hyperparameter $\zeta_2$ | $\{5, 10, 15\}$ |
| Check interval $T$ | 1000 |

### D.6 Implementation Details

Our implementation builds upon the Nanodo training infrastructure (Wortsman et al., 2023), incorporating enhancements for efficiency. This framework relies on Flax (Heek et al., 2023), JAX (Bradbury et al., 2018), and TPUs (Jouppi et al., 2017).

To enable training of larger models, we shard both model and optimizer states, following the methodology of FSDP (Ren et al., 2021), and define these shardings during JIT compilation. Checkpointing is handled using Orbax (Gaffney et al., 2023), while deterministic data loading is facilitated by Grain (Google, 2023).

For data loading, sequences are packed to avoid padding. When a sequence contains fewer tokens than the context length hyperparameter, an end-of-sequence token is appended. This differs from Nanodo (Wortsman et al., 2023), where both begin-of-sequence and end-of-sequence tokens are added.

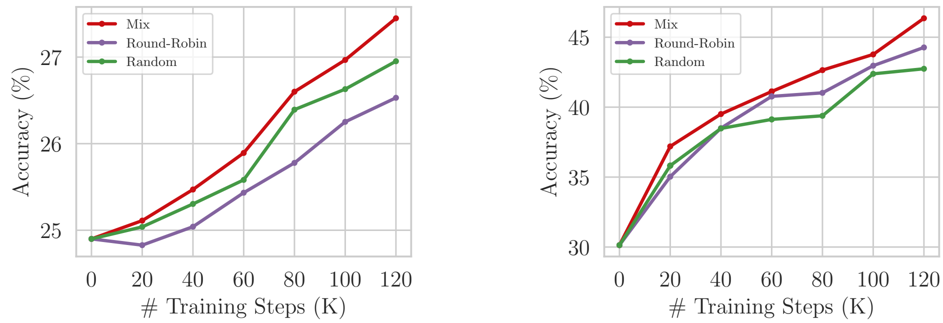
### D.7 Evaluation

Our evaluation adheres to the OLMES suite (Gu et al., 2024). For multilingual downstream performance, we utilize the multilingual version of HellaSwag (Dac Lai et al., 2023), which supports evaluations across 26 languages. English downstream tasks are assessed using ARC-Easy (Clark et al., 2018), CommonsenseQA (Talmor et al., 2018), PIQA (Bisk et al., 2019), and HellaSwag (Zellers et al., 2019). Unless specified otherwise, multilingual evaluations are performed in a 0-shot setting, while GLaM pretraining evaluations employ 7-shot in-context learning, with demonstration candidates separated by two line breaks. For HellaSwag and its translated variants, we evaluate the first 3,000 examples. For all other downstream tasks, evaluations are conducted on their respective validation sets. In the case of multiple-choice tasks, different candidates are included in the prompt, and the average log-likelihood for each candidate is computed. The candidate with the highest score is then selected as the predicted answer.

## E Additional Experiment Results

### E.1 Comparison of Performance Using Mix, Random, and Round-Robin Sampling Strategies

Figure 4 presents the average downstream accuracies of language models pre-trained using Mix, Random, and Round-Robin sampling strategies. In both multilingual pre-training and GLaM pretraining, the Mix sampling strategy consistently outperforms the other two. This motivates us its use in pre-training large language models.




(a): 1B models on multilingual C4 (en), C4 (hi), and C4 (de) datasets

(b): 750M models on GLaM datasets with six domains

Figure 4: Average downstream task accuracy of pretraining language models using Mix, Round-Robin, and Random sampling strategies. Mix and Random use equal batch size for each task ($b_k = b/K, \forall k \in K$).

### E.2 Cosine Similarity and $\underline{c}$-Conflicted Gradients

Figures 5 and 6 show the cosine similarity, defined as $\frac{\langle \mathcal{L}_j(\boldsymbol{\theta}), \mathcal{L}_k(\boldsymbol{\theta}) \rangle}{\|\mathcal{L}_j(\boldsymbol{\theta})\| \|\mathcal{L}_k(\boldsymbol{\theta})\|}$ and the "ratio," defined as $\frac{\langle \mathcal{L}_j(\boldsymbol{\theta}), \mathcal{L}_k(\boldsymbol{\theta}) \rangle}{\|\mathcal{L}_j(\boldsymbol{\theta})\|^2 + \|\mathcal{L}_k(\boldsymbol{\theta})\|^2}$. In particular, if $\frac{\langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \rangle}{\|\mathcal{L}_j(\boldsymbol{\theta})\| \|\mathcal{L}_k(\boldsymbol{\theta})\|} \geq -\tilde{c}$, then the gradients are $\underline{c}$-conflicted for $\underline{c} = \tilde{c}/2$, which aligns with the observations in Figures 5 and 6.

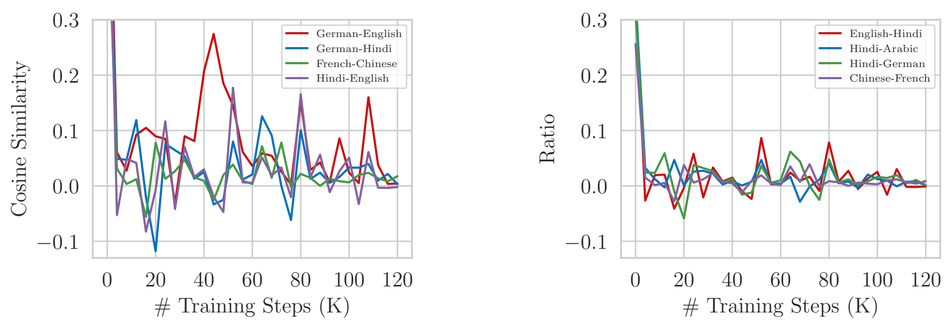



Figure 5: 1B models trained on multilingual mC4 datasets. **Left:** Cosine similarity between task gradients during language model pre-training over time. **Right:** The "ratio," which defined as $\frac{\langle \mathcal{L}_j(\boldsymbol{\theta}), \mathcal{L}_k(\boldsymbol{\theta}) \rangle}{\|\mathcal{L}_j(\boldsymbol{\theta})\|^2 + \|\mathcal{L}_k(\boldsymbol{\theta})\|^2}$, between task gradients during language model pre-training over time. "*data1-data2*" denotes the cosine similarity or ratio between the gradient of *data1* and the gradient of *data2*.

### E.3 Comparison of Performance Using PCGrad, AdaTask, and Mix

Figure 7 presents the average downstream task performance on HellaSwag (en) and HellaSwag (hi) for 270M multilingual language models pre-trained using PCGrad, AdaTask, and Mix. As shown in Figure 7: 1) PCGrad performs similarly to Mix, as it only adjusts gradients when conflicts occur—which is rare. 2) AdaTask converges more slowly due to noisy gradients and suboptimal optimizer state updates. Additionally, both methods are memory-intensive, requiring $O(K)$ storage for task gradients (PCGrad) or optimizer states (AdaTask), making them impractical for large-scale models such as the 540B PaLM (Chowdhery et al., 2022).

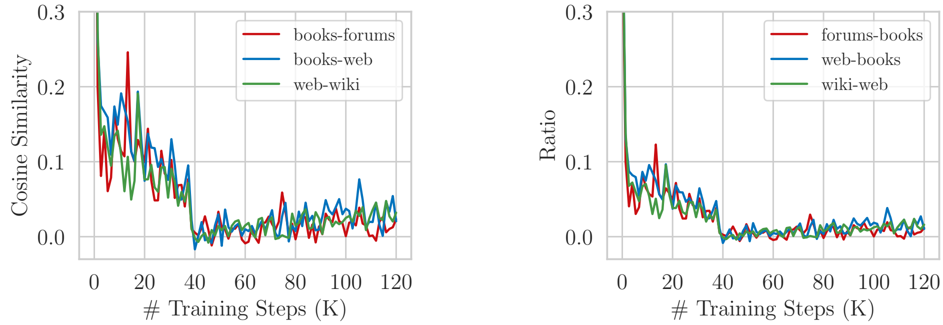



Figure 6: 750M models on GLaM datasets with six domains. **Left:** Cosine similarity between task gradients during language model pre-training over time. **Right:** The "ratio," which defined as $\frac{\langle \mathcal{L}_j(\boldsymbol{\theta}), \mathcal{L}_k(\boldsymbol{\theta}) \rangle}{\|\mathcal{L}_j(\boldsymbol{\theta})\|^2 + \|\mathcal{L}_k(\boldsymbol{\theta})\|^2}$, between task gradients during language model pre-training over time. "*data1-data2*" denotes the cosine similarity or ratio between the gradient of *data1* and the gradient of *data2*.

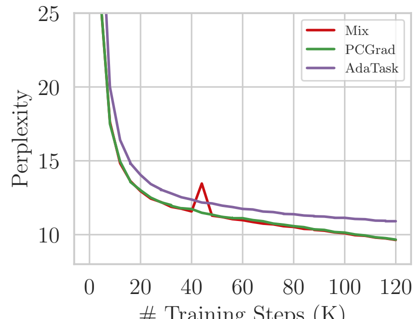



Figure 7: Eval perplexity of pretraining 270M GPT-2 style multilingual language models on mC4 datasets (English and Hindi) using Mix, PCGrad, and AdaTask.

### E.4 Pre-training Results

Tables 8 and 9 present the complete results of pre-training language models across various scales (110M, 270M, 750M, and 1B) and scenarios (Multilingual and GLaM datasets). PiKE consistently outperforms all baselines across all scales and scenarios.

### E.5 Adaptive Sampling Weights of PiKE During Pre-training

Figure 8 illustrates how the adaptive sampling weights of PiKE evolve during language model pre-training. Compared to the Mix sampling strategy, which assigns equal sampling weights to each task, PiKE adaptively adjusts the sampling weights among English, German, and Hindi by leveraging the positive interaction of task gradients. This adaptive data selection allows PiKE to achieve superior performance compared to fixed or heuristic-based baselines.

## F Derivations and Proofs

### F.1 Detailed Derivation of equation (3)

Recall that

$$\mathbf{g}_t = \frac{1}{b_1 + b_2} \left( b_1 e_1 e_1^\top + b_2 e_2 e_2^\top \right) \boldsymbol{\theta}_t + \mathbf{z},$$

Table 8: We report the perplexities (lower the better) on the validation split of multilingual C4 datasets. We also compare the accuracies (%, higher the better) of different models on HellaSwag and its corresponding translated version. HellaSwag and its translated versions have 4 choices. **Bolding** indicates the best model in the task, Metrics means the average across different tasks.

| | | C4 (en) | C4 (hi) | C4 (de) | | HellaSwag (en) | HellaSwag (hi) | HellaSwag (de) |
|---|---|---|---|---|---|---|---|---|
| | Perplexity ↓ | Perplexity ↓ | Perplexity ↓ | Perplexity ↓ | Accuracy(%) ↑ | 0-shot ↑ | 0-shot ↑ | 0-shot ↑ |
| **Single dataset, GPT-2 small style, 270M params, 12 layers default, 120K training steps** | | | | | | | | |
| C4 (en) | 13.25 | 13.25 | * | * | 26.5 | 26.5 | * | * |
| C4 (hi) | 4.97 | * | 4.97 | * | 26.4 | * | 26.4 | * |
| C4 (de) | 11.27 | * | * | 11.27 | 26.1 | * | * | 26.1 |
| **C4 (en) and C4 (hi) datasets, GPT-2 small style, 270M params, 12 layers default, 120K training steps** | | | | | | | | |
| Mix | 10.50 | 15.46 | **5.55** | * | 25.5 | 24.4 | 26.5 | * |
| Round-Robin | 10.57 | 15.57 | 5.57 | * | 25.6 | 25.2 | 26.0 | * |
| Random | 10.57 | 15.57 | 5.57 | * | 25.3 | 24.3 | 26.3 | * |
| PiKE | **10.15** | **14.31** | 5.99 | * | **26.5** | **26.0** | **27.0** | * |
| **C4 (en), C4 (hi), and C4 (de) datasets, GPT-2 small style, 300M params, 12 layers default, 120K training steps** | | | | | | | | |
| Mix | **12.00** | 16.30 | **5.88** | **13.83** | 25.3 | 24.4 | 26.0 | 25.5 |
| Round-Robin | 12.10 | 16.44 | 5.91 | 13.95 | 25.1 | 24.3 | 26.0 | **24.9** |
| Random | 12.16 | 16.49 | 5.95 | 14.03 | 25.1 | 24.7 | **26.6** | 23.9 |
| PiKE | 12.01 | **15.48** | 5.92 | 14.64 | **25.6** | **25.4** | 26.4 | 24.8 |
| **Single dataset, GPT-2 large style, 1B params, 36 Layers default, 120K training steps** | | | | | | | | |
| C4 (en) | 9.30 | 9.30 | * | * | 33.6 | 33.6 | * | * |
| C4 (hi) | 3.87 | * | 3.87 | * | 27.5 | * | 27.5 | * |
| C4 (de) | 7.72 | * | * | 7.72 | 28.1 | * | * | 28.1 |
| **C4 (en) and C4 (hi) datasets, GPT-2 large style, 1B params, 36 Layers default, 120K training steps** | | | | | | | | |
| Mix | 7.41 | 10.60 | **4.22** | * | 27.3 | 28.2 | 26.5 | * |
| Round-Robin | 7.49 | 10.72 | 4.25 | * | 27.5 | 28.0 | 27.0 | * |
| Random | 7.52 | 10.76 | 4.28 | * | 28.0 | 28.9 | 27.0 | * |
| PiKE | **7.21** | **9.63** | 4.80 | * | **30.0** | **32.7** | **27.3** | * |
| **C4 (en), C4 (hi), and C4 (de) datasets, GPT-2 large style, 1B params, 36 Layers default, 120K training steps** | | | | | | | | |
| Mix | **8.29** | 11.13 | **4.45** | **9.29** | 27.5 | 28.1 | 27.1 | **27.6** |
| Round-Robin | 8.41 | 11.31 | 4.97 | 9.46 | 26.5 | 27.6 | 26.7 | 26.3 |
| Random | 8.48 | 11.38 | 4.54 | 9.55 | 26.6 | 27.0 | 26.9 | 26.1 |
| PiKE | 9.56 | **9.49** | 5.32 | 13.87 | **28.7** | **33.0** | **27.2** | 26.2 |

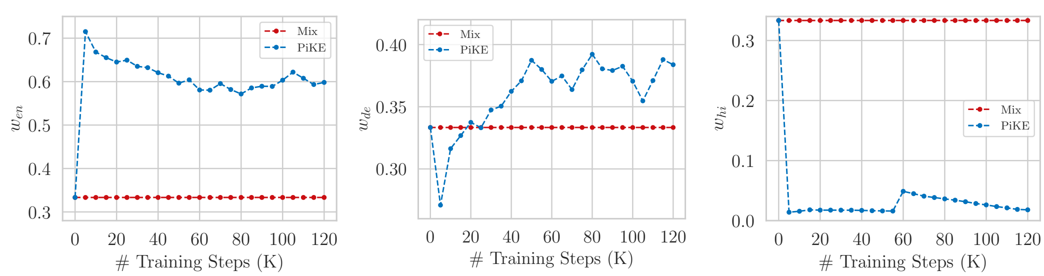



Figure 8: The sampling weights for each dataset during the pre-training of 1B GPT-2-style multilingual language models on mC4 (English), mC4 (Hindi), and mC4 (German). Here, $w_{\text{en}}$ represents the sampling weight for the English dataset, $w_{\text{hi}}$ for the Hindi dataset, and $w_{\text{de}}$ for the German dataset.

Then

$$\begin{aligned}\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t - \eta \frac{1}{b_1+b_2}\left(b_1 e_1 e_1^\top + b_2 e_2 e_2^\top\right)\boldsymbol{\theta}_t - \eta \mathbf{z} \\ &= \boldsymbol{\theta}_t - \frac{\eta}{b}\begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix}\boldsymbol{\theta}_t - \eta\mathbf{z}\end{aligned}$$

Now consider the loss functions for task 1, $\mathcal{L}_1(\theta_{t+1})$, and task 2, $\mathcal{L}_2(\theta_{t+1})$, separately, taking the expectation over the randomness of $\mathbf{z}$

$$\begin{aligned}\mathbb{E}[\mathcal{L}_1(\boldsymbol{\theta}_{t+1})]) &= \mathbb{E}\left[\frac{1}{2}(\mathbf{e}_1^\top \boldsymbol{\theta}_{t+1})^2\right] \\ &= \mathbb{E}\left[\frac{1}{2}\left(\mathbf{e}_1^\top \begin{bmatrix} 1-\frac{\eta b_1}{b} & 0 \\ 0 & 1-\frac{\eta b_2}{b}\end{bmatrix}\boldsymbol{\theta}_t - \mathbf{e}_1^\top \eta \mathbf{z}\right)^2\right] \\ &= \frac{1}{2}\left(\begin{bmatrix}1-\frac{\eta b_1}{b} & 0\end{bmatrix}\boldsymbol{\theta}^\top\right)^2 + \frac{1}{2}\eta^2 \mathbf{e}_1^\top \mathbf{Q}\mathbf{e}_1\end{aligned}$$

Table 9: We report perplexity (lower is better) on the validation split of the GLaM datasets, averaging perplexities across six domains when applicable or reporting a single perplexity when only training with a single domain. We also compare the accuracies (%, higher the better) of different models on four different Q/A tasks. HellaSwag and ArcE tasks have 4 choices, CSQA has 5 choices, and PIQA has 2 choices. PiKE (Uniform) means PiKE using initial sampling weights of $1/6$ for each task and PiKE (GLaM) means PiKE using GLaM tuned weights as initial task weights. **Bolding** indicates the best model in the task, $\overline{\text{Metrics}}$ means the average across different tasks, underlining indicates PiKE beating Mix, Round-Robin, Random methods

| | GLaM | | ArcE | CSQA | HellaSwag | PIQA |
|---|---|---|---|---|---|---|
| | $\overline{\text{Perplexity}}\downarrow$ | $\overline{\text{Accuracy(\%)}}\uparrow$ | 7-shot $\uparrow$ | 7-shot $\uparrow$ | 7-shot $\uparrow$ | 7-shot $\uparrow$ |
| **Single domain of GLaM dataset, GPT-2 small style, 110M params, 12 layers default** | | | | | | |
| Wikipedia | 9.96 | 33.5 | 32.5 | 20.9 | 27.3 | 53.3 |
| Filtered Webpage | 16.05 | 37.2 | 38.4 | 26.8 | 27.6 | 55.8 |
| News | 9.33 | 33.8 | 31.1 | 22.7 | 27.0 | 54.5 |
| Forums | 22.87 | 35.5 | 32.1 | 23.4 | 28.7 | 57.6 |
| Books | 16.81 | 34.7 | 34.3 | 22.1 | 27.8 | 54.7 |
| Conversations | 18.27 | 36.1 | 32.6 | 25.6 | 28.6 | 57.6 |
| **Six domains of GLaM dataset, GPT-2 small style, 110M params, 12 layers default** | | | | | | |
| Mix | **18.27** | 36.2 | 35.6 | 24.1 | **28.5** | 56.7 |
| Round-Robin | 18.45 | 35.9 | 35.8 | 24.2 | 27.5 | 56.0 |
| Random | 18.48 | 35.5 | 34.3 | 22.4 | 28.4 | 56.8 |
| GLaM | 18.91 | 35.8 | 35.3 | 24.1 | 28.5 | 55.1 |
| DoReMi | 18.98 | 37.0 | 36.0 | **28.3** | 28.2 | 55.3 |
| PiKE (Uniform) | 18.44 | 37.4 | 36.8 | 27.5 | **28.5** | **57.0** |
| PiKE (GLaM) | 19.34 | **37.8** | **39.0** | 27.0 | 28.0 | **57.0** |
| **Single domain of GLaM dataset, GPT-2 large style, 750M params, 36 layers default** | | | | | | |
| Wikipedia | 7.24 | 35.9 | 35.1 | 24.0 | 30.5 | 53.9 |
| Filtered Webpage | 11.12 | 40.9 | 36.7 | 33.2 | 34.2 | 56.5 |
| News | 6.62 | 37.4 | 33.6 | 24.7 | 34.1 | 57.3 |
| Forums | 16.29 | 43.6 | 38.0 | 35.8 | 39.7 | 60.7 |
| Books | 11.83 | 41.3 | 40.0 | 33.0 | 34.5 | 57.8 |
| Conversations | 13.50 | 42.2 | 36.9 | 33.2 | 39.2 | 59.6 |
| **Six domains of GLaM dataset, GPT-2 large style, 750M params, 36 layers default** | | | | | | |
| Mix | **12.77** | 46.4 | 47.2 | 39.6 | 37.9 | 60.9 |
| Round-Robin | 12.98 | 44.3 | 43.5 | 36.7 | 36.8 | 60.3 |
| Random | 12.99 | 42.7 | 41.7 | 34.2 | 36.6 | 58.2 |
| GLaM | 13.20 | 45.3 | 46.9 | 39.8 | **38.0** | 56.4 |
| DoReMi | 13.25 | 46.5 | 48.6 | 40.1 | 37.5 | 59.6 |
| PiKE (Uniform) | 13.22 | 47.6 | 49.6 | 43.2 | 37.2 | 60.4 |
| PiKE (GLaM) | 13.35 | **48.1** | **49.8** | **43.5** | **38.0** | **61.2** |

$$= \frac{1}{2}\left(\left(1-\frac{\eta b_1}{b}\right)\theta_{1,t}\right)^2 + \frac{1}{2}\eta^2 \mathbf{e}_1^\top \mathbf{Q}\mathbf{e}_1$$

Similarly, for task 2, we have

$$\mathbb{E}[\mathcal{L}_2(\boldsymbol{\theta}_{t+1})]) = \frac{1}{2}\left(\left(1-\frac{\eta b_2}{b}\right)\theta_{2,t}\right)^2 + \frac{1}{2}\eta^2 \mathbf{e}_2^\top \mathbf{Q}\mathbf{e}_2$$

where $\theta_{1,t}$ and $\theta_{2,t}$ denote the first and second component of the vector $\boldsymbol{\theta}_t$. Combining the losses for both tasks, the total expected loss becomes

$$\begin{aligned}
\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_{t+1})] &= \mathbb{E}[\mathcal{L}_1(\boldsymbol{\theta}_{t+1})]) + \mathbb{E}[\mathcal{L}_2(\boldsymbol{\theta}_{t+1})]) \\
&= \frac{1}{2}\left(\left(1-\frac{\eta b_1}{b}\right)\theta_{1,t}\right)^2 + \frac{1}{2}\left(\left(1-\frac{\eta b_2}{b}\right)\theta_{2,t}\right)^2 + \eta^2\frac{b_1\sigma_1^2 + b_2\sigma_2^2}{b^2} \\
&= \frac{1}{2}(1-\frac{\eta b_1}{b})^2\theta_{1,t}^2 + \frac{1}{2}(1-\frac{\eta b_2}{b})^2\theta_{2,t}^2 + \eta^2\frac{b_1\sigma_1^2 + b_2\sigma_2^2}{b^2},
\end{aligned}$$

which completes the derivations.

### F.2 PiKE: Main Theoretical Results

**Lemma F.1.** *Assume* $\frac{1}{2(K-1)} > \underline{c}$*. If* $\|\nabla \mathcal{L}(\boldsymbol{\theta})\|^2 \leq \epsilon$*, we have*

$$\sum_{k=1}^{K} \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 \leq \frac{\epsilon}{1 - 2\,\underline{c}\,(K-1)}.$$

*Conversely, if* $\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 \leq \delta_k,\ \forall k$*, then*

$$\|\nabla \mathcal{L}(\boldsymbol{\theta})\|^2 \leq (1-\bar{c}) \sum_{k=1}^{K} \delta_k \ + \ \bar{c} \left( \sum_{k=1}^{K} \sqrt{\delta_k} \right)^2$$

*Proof:* We first prove the first direction. Notice that

$$\begin{aligned} \|\nabla \mathcal{L}(\boldsymbol{\theta})\|^2 &= \| \sum_{k=1}^{K} \nabla \mathcal{L}_k(\boldsymbol{\theta}) \|^2 \\ &= \sum_{k=1}^{K} \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} \sum_{j \neq k} \langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \rangle \leq \epsilon \end{aligned}$$

where we use the definition of $\nabla \mathcal{L}(\boldsymbol{\theta})$ and expand the term. Then we have

$$\begin{aligned} \sum_{k=1}^{K} \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} \sum_{j \neq k} \langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \rangle &\overset{(a)}{\geq} \sum_{k=1}^{K} \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 - \underline{c} \sum_{k=1}^{K} \sum_{j \neq k} \left( \|\nabla \mathcal{L}_j(\boldsymbol{\theta})\|^2 + \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 \right) \\ &\overset{(b)}{\geq} \sum_{k=1}^{K} \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 \left(1 - 2\underline{c}(K-1)\right) \end{aligned}$$

where $(a)$ uses the Definition 3.2, $(b)$ uses symmetric identity. Thus we get

$$\sum_{k=1}^{K} \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 \leq \frac{\epsilon}{1 - 2\,\underline{c}\,(K-1)}$$

This completes the proof of the first inequality. We now prove the second inequality. Notice that

$$\begin{aligned} \|\nabla \mathcal{L}(\boldsymbol{\theta})\|^2 = \| \sum_{k=1}^{K} \nabla \mathcal{L}_k(\boldsymbol{\theta}) \|^2 &= \sum_{k=1}^{K} \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} \sum_{j \neq k} \langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \rangle \\ &\overset{(a)}{\leq} \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \bar{c} \sum_{k=1}^{K} \sum_{j \neq k} \|\nabla \mathcal{L}_j(\boldsymbol{\theta})\|^2 \|\mathcal{L}_k(\boldsymbol{\theta})\|^2 \\ &= (1-\bar{c}) \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \bar{c} \|\nabla \mathcal{L}_k\|^2 + \bar{c} \sum_{k=1}^{K} \sum_{j \neq k} \|\nabla \mathcal{L}_j(\boldsymbol{\theta})\|^2 \|\mathcal{L}_k(\boldsymbol{\theta})\|^2 \\ &\overset{(b)}{\leq} (1-\bar{c}) \sum_{k=1}^{K} \delta_k + \bar{c} \left( \sum_{k=1}^{K} \sqrt{\delta_k} \right)^2 \end{aligned}$$

where $(a)$ use the Definition 3.3 and $(b)$ combines the second and third terms and use the condition that $\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 \leq \delta_k$. This completes the proof of the second inequality.

**Lemma F.2.** *For the optimization problem*

$$\begin{aligned} \min_{w_1,\ldots,w_K} \quad & \sum_{k=1}^{K} w_k \lambda_k + \frac{1}{2} w_k^2 \kappa_k \\ \textit{s.t.} \quad & \sum_{k=1}^{K} w_k = 1, \quad w_k \geq 0, \quad \forall k \end{aligned} \tag{13}$$

*the optimal solution is*

$$w_k^* = \max\left\{0, -\frac{\mu + \lambda_k}{\kappa_k}\right\} \tag{14}$$

*where $\mu$ is chosen such that $\sum_{k=1}^{K} w_k^* = 1$*

*Proof:* Consider the Lagrangian function

$$\mathcal{L}(w_1, \ldots, w_k, \mu, \alpha_1, \ldots, \alpha_k) = \sum_{k=1}^{K} w_k \lambda_k + \frac{1}{2} w_k^2 \kappa_k + \mu \left( \sum_{k=1}^{K} w_k - 1 \right) - \sum_{k=1}^{K} \alpha_k w_k$$

where $\mu$ is Lagrange multiplier for the equality constraint for the constraint $\sum_{k=1}^{K} w_k = 1$ and $\alpha_K \geq 0$ are Lagrange multipliers for the nonnegativity constraints $w_k$. Take the partial derivative of $\mathcal{L}$ with respect to $w_k$ and set it to 0:

$$\frac{\partial \mathcal{L}}{\partial w_k} = \lambda_k + w_k \kappa_k + \mu - \alpha_k = 0$$

From the Karush-Kuhn-Tucker (KKT) conditions, we also have $w_k^\star \geq 0, \alpha_k \geq 0$, and $\alpha_k w_k^\star = 0$. If $w_k^\star > 0$, then $\alpha_k = 0$, which implies

$$0 = \lambda_k + w_k^\star \kappa_k + \mu \quad \Longrightarrow \quad w_k^\star = -\frac{\mu + \lambda_k}{\kappa_k}$$

If $-(\mu + \lambda_k)/\kappa_k$ is negative, then $w_k^\star = 0$ must hold. Combining these, we get

$$w_k^* = \max\left\{0, -\frac{\mu + \lambda_k}{\kappa_k}\right\}$$

Finally, the Lagrange multiplier $\mu$ is determined by enforcing the equality constraint:

$$\sum_{k=1}^{K} w_k^* = 1$$

with $\mu$ chosen so that the $w_k^*$ sum to 1 . This completes the proof.

**Theorem F.3.** *(Theorem 3.5 in the main body) Suppose Assumption 3.4 is satisfied. Assume that at the given point $\boldsymbol{\theta}_t$ the gradients are $\underline{c}$-conflicted and $\bar{c}$-aligned with $\underline{c} < \frac{1}{K-2+b/b_k}, \forall k$. Moreover, assume the gradient is computed according to the mix strategy equation (2). Then, we have*

$$\mathbb{E}[\mathcal{L}(\boldsymbol{\theta} - \eta \mathbf{g})] \leq \mathcal{L}(\boldsymbol{\theta}) + \sum_{k=1}^{K} b_k \Big( -\frac{\eta}{b} \beta \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \frac{L\eta^2}{2b^2} \sigma_k^2 \Big) + \sum_{k=1}^{K} b_k^2 \frac{L\eta^2}{2b^2} \gamma \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 \tag{15}$$

*where $0 \leq \beta \triangleq \min_k(1 + \underline{c}(-K + 2 - \frac{b}{b_k}))$ and $\gamma \triangleq 1 + \bar{c}(K-1)$.*

*Proof:* We begin by revisiting the multi-task optimization problem under consideration. The objective is defined as:

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \mathcal{L}(\boldsymbol{\theta}) := \sum_{k=1}^{K} \mathbb{E}_{x \sim \mathcal{D}_k} \left[ \ell_k(\boldsymbol{\theta}; x) \right], \tag{16}$$

where $\mathcal{L}(\boldsymbol{\theta})$ is the expected aggregate loss over all tasks. Assume we mix the gradients with taking $b_k$ i.i.d. samples from task $k$ for $k = 1, \ldots, K$. Then under the Assumption 3.4 the estimated gradient direction is given by

$$
\begin{aligned}
\mathbf{g} &= \frac{1}{\sum_{k=1}^{K} b_k} \left( \sum_{k=1}^{K} \sum_{\substack{i=1 \\ x_i \sim \mathcal{D}_k}}^{b_k} \nabla \ell_k(\boldsymbol{\theta}; x_i) \right) \\
&= \frac{1}{b} \sum_{k=1}^{K} \left( b_k \nabla \ell_k(\boldsymbol{\theta}) \right) \ + \ \mathbf{z},
\end{aligned} \tag{17}
$$

where the random variable $\mathbf{z}$ is defined as $\mathbf{z} = \sum_{k=1}^{K} \sum_{i=1, x_i \sim \mathcal{D}_k}^{b_k} (\nabla \ell_k(\boldsymbol{\theta}, x_i) - \nabla \mathcal{L}_k(\boldsymbol{\theta}))$ over the randomness of the sampling strategy. Let $\boldsymbol{\theta}^+$ be the updated point after gradient descent with $\boldsymbol{\theta}^+ = \boldsymbol{\theta} - \eta \mathbf{g}$. By the descent lemma, the following inequality holds for the updated parameter $\boldsymbol{\theta}^+$:

$$
\mathcal{L}(\boldsymbol{\theta}^+) \leq \mathcal{L}(\boldsymbol{\theta}) - \eta \mathbf{g}^\top \nabla \mathcal{L}(\boldsymbol{\theta}) + \frac{L\eta^2}{2} \|\mathbf{g}\|^2, \tag{18}
$$

Taking the expectation over the randomness of $\mathbf{z}$, we obtain:

$$
\begin{aligned}
\mathbb{E}\left[\mathcal{L}(\boldsymbol{\theta}^+)\right] &\leq \mathcal{L}(\boldsymbol{\theta}) - \eta \mathbb{E}[\mathbf{g}]^\top \nabla \mathcal{L}(\boldsymbol{\theta}) + \frac{L\eta^2}{2} \mathbb{E}\left(\|\mathbf{g}\|^2\right) \\
&\overset{(a)}{=} \mathcal{L}(\boldsymbol{\theta}) - \eta \left( \frac{1}{b} \sum_{k=1}^{K} b_k \nabla \mathcal{L}_k(\boldsymbol{\theta}) \right)^\top \left( \sum_{k=1}^{K} \nabla \mathcal{L}_k(\boldsymbol{\theta}) \right) \\
&\quad + \frac{L\eta^2}{2b^2} \left( \left( \sum_{k=1}^{K} b_k \nabla \mathcal{L}_k(\boldsymbol{\theta}) \right)^2 + \sum_{k=1}^{K} (b_k \sigma_k^2) \right) \\
&\overset{(b)}{=} \mathcal{L}(\boldsymbol{\theta}) - \frac{\eta}{b} \left( \sum_{k=1}^{K} b_k \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} \sum_{j \neq k} b_k \left\langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \right\rangle \right) \\
&\quad + \frac{L\eta^2}{2b^2} \left( \left( \sum_{k=1}^{K} b_k \nabla \mathcal{L}_k(\boldsymbol{\theta}) \right)^2 + \sum_{k=1}^{K} (b_k \sigma_k^2) \right),
\end{aligned}
$$

where $(a)$ substitutes the definition of $\mathbf{g}$ and uses the Assumption 3.4, and $(b)$ expands the terms. We have

$$
\begin{aligned}
\mathbb{E}\left[\mathcal{L}(\boldsymbol{\theta}^+)\right] &\overset{(a)}{\leq} \mathcal{L}(\boldsymbol{\theta}) - \frac{\eta}{b} \left( \sum_{k=1}^{K} b_k \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 - \sum_{k=1}^{K} \sum_{j \neq k} b_k \underline{c} (\|\nabla \mathcal{L}_j(\boldsymbol{\theta})\|^2 + \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2) \right) \\
&\quad + \frac{L\eta^2}{2b^2} \left( \sum_{k=1}^{K} b_k^2 \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} \sum_{j \neq k} b_j b_k \left\langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \right\rangle + \sum_{k=1}^{K} b_k \sigma_k^2 \right), \\
&\overset{(b)}{=} \mathcal{L}(\boldsymbol{\theta}) - \frac{\eta}{b} \left( \sum_{k=1}^{K} \left( b_k - \underline{c} b_k (K-1) - \underline{c} \sum_{j \neq k} b_j \right) \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 \right) \\
&\quad + \frac{L\eta^2}{2b^2} \left( \sum_{k=1}^{K} b_k^2 \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} \sum_{j \neq k} b_j b_k \left\langle \nabla \mathcal{L}_j(\boldsymbol{\theta}), \nabla \mathcal{L}_k(\boldsymbol{\theta}) \right\rangle + \sum_{k=1}^{K} b_k \sigma_k^2 \right), \\
&\overset{(c)}{\leq} \mathcal{L}(\boldsymbol{\theta}) - \frac{\eta}{b} \left( \sum_{k=1}^{K} (b_k - \underline{c}(K-1) b_k - \underline{c}(b - b_k)) \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 \right) \\
&\quad + \frac{L\eta^2}{2b^2} \left( \sum_{k=1}^{K} b_k^2 \|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} \sum_{j \neq k} \bar{c} b_j b_k \|\nabla \mathcal{L}_j(\boldsymbol{\theta})\|^2 \|\nabla \mathcal{L}_j(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} b_k \sigma_k^2 \right)
\end{aligned}
$$

$$
\begin{aligned}
&\overset{(d)}{=} \mathcal{L}(\boldsymbol{\theta}) - \frac{\eta}{b}\left(\sum_{k=1}^{K} b_k(1+\underline{c}(-K+2-b/b_k))\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2\right) \\
&\quad + \frac{L\eta^2}{2b^2}\left(\bar{c}\left(\sum_{k=1}^{K} b_k\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|\right)^2 + (1-\bar{c})\sum_{k=1}^{K} b_k^2\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} b_k\sigma_k^2\right) \\
&\overset{(e)}{\leq} \mathcal{L}(\boldsymbol{\theta}) - \frac{\eta}{b}\left(\sum_{k=1}^{K} b_k(1+\underline{c}(-K+2-b/b_k))\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2\right) \\
&\quad + \frac{L\eta^2}{2b^2}\left(\bar{c}K\sum_{k=1}^{K} b_k^2\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + (1-\bar{c})\sum_{k=1}^{K} b_k^2\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} b_k\sigma_k^2\right) \\
&= \mathcal{L}(\boldsymbol{\theta}) - \frac{\eta}{b}\left(\sum_{k=1}^{K} b_k(1+\underline{c}(-K+2-b/b_k))\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2\right) \\
&\quad + \frac{L\eta^2}{2b^2}\left((1-\bar{c}+\bar{c}K)\sum_{k=1}^{K} b_k^2\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} b_k\sigma_k^2\right) \qquad (19)
\end{aligned}
$$

where $(a)$ applies Definition 3.2 to the second term and expands the third term, $(b)$ expands the summation in the second term, $(c)$ uses the identity $\sum_{k=1}^{K}\sum_{j\neq k} b_j = \sum_{k=1}^{K}(b-b_k)$ in the second term and applies Definition 3.3 to the third term, $(d)$ combines terms in the third term, and $(e)$ uses the inequality $\|\sum_{i=1}^{N} u_i\|^2 \leq N\sum_{i=1}^{N} u_i^2$, where $\mathbf{u}$ is a column vector. We define $\beta$ and $\gamma$ such that

$$
\begin{aligned}
\beta &= \min_k(1+\underline{c}(-K+2-\frac{b}{b_k})) \\
\gamma &= 1+\bar{c}(K-1)
\end{aligned} \qquad (20)
$$

Then using the definition of $\beta$ and $\gamma$, substituting back we have

$$
\begin{aligned}
\mathbb{E}\left[\mathcal{L}(\boldsymbol{\theta}^+)\right] &\leq \mathcal{L}(\boldsymbol{\theta}) - \frac{\eta\beta}{b}\left(\sum_{k=1}^{K} b_k\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2\right) + \frac{L\eta^2}{2b^2}\left(\gamma\sum_{k=1}^{K} b_k^2\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \sum_{k=1}^{K} b_k\sigma_k^2\right) \\
&= \mathcal{L}(\boldsymbol{\theta}) + \sum_{k=1}^{K} b_k\left(-\frac{\eta\beta}{b}\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2 + \frac{L\eta^2}{2b^2}\sigma^2\right) + \sum_{k=1}^{K} b_k^2\frac{L\eta^2}{2b^2}\gamma\|\nabla \mathcal{L}_k(\boldsymbol{\theta})\|^2
\end{aligned}
$$

which we complete the proof.

**Theorem F.4.** *(Theorem 3.6 in the main body) Suppose the assumptions in Theorem F.3 is satisfied and we run the Conceptual PiKE Algorithm (Algorithm 2) initialized at $\boldsymbol{\theta}_0$ with the SGD optimizer in Step 10 of the algorithm. Let $\Delta_L = \mathcal{L}(\boldsymbol{\theta}_0) - \min_{\boldsymbol{\theta}} \mathcal{L}(\theta)$ and $\sigma_{\max} = \max_k \sigma_k$. Suppose $\delta > 0$ is a given constant and the stepsize $\eta \leq \frac{\beta\delta}{L\sigma_{\max}^2/b + L\eta\delta}$. Then, after $T = \frac{2\beta\Delta_L}{\eta\delta}$ iterations, Algorithm Algorithm 2 finds a point $\bar{\boldsymbol{\theta}}$ such that*

$$
\mathbb{E}\|\nabla \mathcal{L}_k(\bar{\theta})\|^2 \leq \delta, \quad \forall k = 1, \ldots, K. \qquad (21)
$$

*Moreover, if we choose $\eta = \frac{\beta\delta}{L\sigma_{\max}^2/b + L\eta\delta}$, then the Conceptual PiKE algorithm requires at most*

$$
\bar{T} = \frac{2L\Delta_L(\sigma_{\max}^2/b + \gamma\delta)}{\delta^2\beta^2}
$$

*iterations to find a point satisfying equation (21).*

*Proof:* We prove this by contradiction. Assume that $\max_k \|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t)\|^2 > \delta$ for $t = 0, \ldots, T$. First notice that Theorem F.3 implies that for all t, we have

$$
\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_{t+1})] \leq \mathcal{L}(\boldsymbol{\theta}_t) + \sum_{k=1}^{K} w_k^\star\left(-\eta\beta\|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t))\|^2 + \frac{L\eta^2\sigma_{\max}^2}{2b}\right) + \sum_{k=1}^{K}\frac{w_k^\star}{2}\left(L\eta^2\gamma\|\nabla \mathcal{L}_k(\boldsymbol{\theta}_t)\|^2\right) \qquad (22)
$$

where $\{w_k^\star\}_{k=1}^K$ is the minimizer of the RHS of the equation (22) on the constrained set $\{(w_1, \ldots, w_k) | \sum_{k=1}^K w_k = 1, \; w_k \geq 0 \; \forall k \in K\}$. Since $w_k^\star$ is the minimizer of the RHS of equation (22), we have

$$w_k^\star \left(-\eta\beta\|\nabla\mathcal{L}_k(\boldsymbol{\theta}_t)\|^2 + \frac{L\eta^2}{2b}\sigma_{\max}^2\right) + \frac{w_k^\star}{2} L\eta^2\gamma\|\nabla\mathcal{L}_k(\boldsymbol{\theta}_t)\|^2 \leq \left(-\eta\beta\|\nabla\mathcal{L}_{k_t^\star}(\boldsymbol{\theta}_t)\|^2 + \frac{2\eta^2}{2b}\sigma_{\max}^2\right) + \frac{L\eta^2}{2}\gamma\|\nabla\mathcal{L}_{k_t^\star}(\boldsymbol{\theta}_t)\|^2 \tag{23}$$

where $k_t^\star \in \arg\max_k \|\nabla\mathcal{L}_k(\boldsymbol{\theta}_t)\|^2$. Moreover since

$$\eta \leq \frac{\beta\|\nabla\mathcal{L}_k(\boldsymbol{\theta})\|^2}{L\frac{\sigma_{\max}^2}{b} + L\gamma\|\nabla\mathcal{L}_k(\boldsymbol{\theta}_t)\|^2},$$

we have

$$\left(-\eta\beta\|\nabla\mathcal{L}_{k_t^\star}(\boldsymbol{\theta}_t)\|^2 + \frac{2\eta^2}{2b}\sigma_{\max}^2\right) + \frac{L\eta^2}{2}\gamma\|\nabla\mathcal{L}_{k_t^\star}(\boldsymbol{\theta}_t)\|^2 \leq -\frac{\beta\eta}{2}\|\nabla\mathcal{L}_{k_t^\star}(\boldsymbol{\theta}_t)\|^2 \tag{24}$$

Combining equation (22), (23), and (24), we obtain

$$\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_{t+1})] \leq \mathcal{L}(\boldsymbol{\theta}_t) - \frac{\beta\eta}{2}\|\nabla\mathcal{L}_{k_t^\star}(\boldsymbol{\theta}_t)\|^2$$

Or equivalently

$$\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_{t+1})] \leq \mathcal{L}(\boldsymbol{\theta}_t) - \frac{\beta\eta}{2}\max_k\|\nabla\mathcal{L}_k(\boldsymbol{\theta}_t)\|^2$$

Summing the above inequality from $t = 0$ to $t = T - 1$, we get

$$\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_T)] \leq \mathcal{L}(\boldsymbol{\theta}_0) - \mathbb{E}\frac{\beta\eta}{2}\sum_{t=1}^{T-1}\max_k\|\nabla\mathcal{L}_k(\boldsymbol{\theta}_t)\|^2$$

According to the contradiction assumption, we get

$$\mathbb{E}[\mathcal{L}(\boldsymbol{\theta}_T)] \leq \mathcal{L}(\boldsymbol{\theta}_0) - \frac{\beta\eta}{2}T\delta$$

Using the definition $\Delta_{\mathcal{L}} \triangleq \mathcal{L}(\boldsymbol{\theta}_0) - \min_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta})$, we get

$$T \leq \frac{2\Delta_{\mathcal{L}}}{\beta\eta\delta}$$

Finally notice that by setting $\eta = \frac{\beta\delta}{L\frac{\sigma_{\max}^2}{b} + L\gamma\delta}$, we get

$$T \leq \bar{T} = \frac{2\Delta_{\mathcal{L}}}{\beta\eta} = \frac{2L\Delta_{\mathcal{L}}}{\beta\delta^2}\left(\frac{\sigma_{\max}^2}{b} + \gamma\delta\right)$$

which means after iteration $T$ steps, we have

$$\min_t\left\{\max_k\|\nabla\mathcal{L}_k(\boldsymbol{\theta}_t)\|^2\right\} \leq \delta,$$

which completes the proof.

### F.3 PIKE: FAIRNESS RELATED RESULTS

Consider the tilted empirical risk minimization (Li et al., 2020):

$$\min_{\boldsymbol{\theta}} \quad \widetilde{\mathcal{L}}(\tau;\boldsymbol{\theta}) := \frac{1}{\tau}\log\left(\sum_{k=1}^K e^{\tau\mathcal{L}_k(\boldsymbol{\theta})}\right).$$

As we described in the main body, we connect this problem to the minimization of the weighted sum of $\mathcal{L}_k$'s using the following lemma:

**Lemma F.5.** *(Lemma 3.7 in the main body) Let $\mathbf{x} \in \mathbb{R}^K$ and $\tau > 0$. Then*

$$\log\left(\sum_{k=1}^K e^{\tau x_k}\right) = \max_{\substack{\mathbf{y}\in\mathbb{R}_+^K \\ \sum_{k=1}^K y_k = \tau}} \left(\sum_{k=1}^K y_k x_k - \sum_{k=1}^K \frac{y_k}{\tau}\log\left(\frac{y_k}{\tau}\right)\right)$$

*Proof:* Let

$$f(\mathbf{x}) = \log\left(\sum_{k=1}^{K} e^{\tau x_k}\right)$$

Then, the conjugate dual of the function $f(\cdot)$ can be computed as

$$f^{\star}(\mathbf{y}) = \sup_{\mathbf{x}} \left(\sum_{k=1}^{K} x_k y_k - \log\left(\sum_{k=1}^{K} e^{\tau x_k}\right)\right)$$

Taking the partial derivative of the objective with respect to $x_i$ and setting it to zero gives

$$x_k^{\star} = \frac{1}{\tau}\log\left(\frac{\phi}{\tau}\right) + \frac{1}{\tau}\log(y_k)$$

where $\phi \triangleq \sum_{k=1}^{K} e^{\tau x_k}$. Substituting the optimal value of $x_k^{\star}$, we get

$$\begin{aligned}
f^{\star}(\mathbf{y}) &= \sum_{k=1}^{K} y_k \left(\frac{1}{\tau}\log\left(\frac{\phi}{\tau}\right) + \frac{1}{\tau}\log y_k\right) - \log\left(\sum_{k=1}^{K} \frac{\phi y_k}{\tau}\right) \\
&= \sum_{k=1}^{K} \frac{y_k}{\tau}\log\left(\frac{\phi}{\tau}\right) + \sum_{k=1}^{K} \frac{y_k}{\tau}\log(y_k) - \log\left(\sum_{k=1}^{K} \frac{\phi y_k}{\tau}\right) \\
&\overset{(a)}{=} \log\left(\frac{\phi}{\tau}\right) + \sum_{k=1}^{K} \frac{y_k}{\tau}\log(y_k) - \log(\phi) \\
&= -\log(\tau) + \sum_{k=1}^{K} \frac{y_k}{t}\log y_k \\
&= \sum_{k=1}^{K} \frac{y_k}{\tau}\log\left(\frac{y_k}{\tau}\right)
\end{aligned}$$

where $(a)$ uses the condition that $\sum_{k=1}^{K} y_k = \tau$. We apply Fenchel's duality theorem again, and then we have

$$f(\mathbf{x}) = f^{\star\star}(\mathbf{x}) = \max_{\substack{\mathbf{y}\in\mathbb{R}^K \\ \sum_{k=1}^{K} y_k = \tau}} \left(\sum_{k=1}^{K} y_k x_k - \sum_{k=1}^{K} \frac{y_k}{\tau}\log\left(\frac{y_k}{\tau}\right)\right),$$

which completes the proof.

**Lemma F.6.** *For the problem*

$$\max_{\substack{\mathbf{y}\in\mathbb{R}_+^K \\ \sum_{k=1}^{K} y_k = \tau}} \left(\sum_{k=1}^{K} y_k x_k - \sum_{k=1}^{K} \frac{y_k}{\tau}\log\left(\frac{y_k}{\tau}\right)\right),$$

*the optimal* $\mathbf{y}$ *is given by*

$$y_k^{\star} = \frac{\tau e^{\tau x_k - 1}}{\sum_{k=1}^{K} e^{\tau x_k - 1}}$$

*Proof:* We start by forming and maximizing the Lagrangian function

$$\max_{\mathbf{y}\in\mathbb{R}^K} \left(\sum_{k=1}^{K} y_k x_k - \sum_{k=1}^{K} \frac{y_k}{\tau}\log\left(\frac{y_k}{\tau}\right) + \mu\left(\sum_{k=1}^{K} y_k - \tau\right)\right)$$

where $\mu$ is a free variable. Taking the partial derivative of the objective with respect to $y_k$ and setting it to zero gives

$$y_k^{\star} = \alpha\tau e^{\tau x_k - 1},$$

where the coefficient $\alpha$ should be chosen such that $\sum_k y_k^* = 1$, implying

$$y_k^{\star} = \frac{\tau e^{\tau x_k - 1}}{\sum_{k=1}^{K} e^{\tau x_k - 1}}.$$