A SUPPLEMENTAL RELATED WORKS

Block coordinate descent Block Coordinate Descent (BCD) involves iteratively optimizing over a block of coordinates while holding the others constant. The foundational work of Tseng (2001) provides a comprehensive analysis of the convergence properties of BCD under certain conditions. Subsequent research has explored various BCD variants (Hong et al., 2017), including randomized BCD (Nesterov, 2012; Richtárik & Takáč, 2014; Lu & Xiao, 2015), cyclic BCD (Sun & Hong, 2015), and greedy BCD (Nutini et al., 2015). Among these, the greedy variant, also known as Gauss-Southwell BCD method, has drawn attention due to its ability to prioritize coordinates that yield the most substantial improvement in each iteration, thereby potentially accelerating convergence.

In the realm of machine learning, BCD has also found applications (Nutini et al., 2022). For example, Luo et al. (2024) leverages BCD to perform memory-efficient fine-tuning of LLM and Xu & Zhang (2024) uses random masking to perform this. In federated learning, Rothchild et al. (2020) adopts top-k momentum value unsketch rather than our top-k momentum filtering to tackle communication bottleneck and convergence issues. In LLMs, some concurrent works propose BCD-based algorithms leveraging task vectors to enhance fine-tuning performance (Li et al., 2024) and mitigate catastrophic forgetting in multi-task learning (Panda et al., 2024). In a recent work (Hui et al., 2024), catastrophic forgetting during the fine-tuning of LLMs is addressed by selectively freezing 50% of the model parameters during training. Our approach is akin to a more efficient greedy BCD, achieving superior performance in fine-tuning tasks and alleviating forgetting better.

B Supplementary Analysis on the Top-$\alpha\%$ Filter

In this section, we provide supplementary analysis on our top-$\alpha\%$ filter, which serves as a preliminary for proving Theorem 1 in Appendix C.

As introduced in Section D.4, the entire parameter space is divided into B parts, with the k-th part having a dimension of d_k. We assume the parameter space is $\mathbb{R}^d$, which can be expressed as the product $\mathbb{R}^d \cong \mathbb{R}^{d_1} \times \mathbb{R}^{d_2} \times \cdots \times \mathbb{R}^{d_B}$. For any $z \in \mathbb{R}^d$, we represent it as:

$$z = \text{Concat}(z^{(1)}, z^{(2)}, \ldots, z^{(B)}),$$

where $z^{(k)} \in \mathbb{R}^{d_k}$ for each $1 \leq k \leq B$.

Definition 1. *For any $z \in \mathbb{R}^d$, we define the top-$\alpha\%$ filter of z as*

$$\texttt{FLT}_\alpha(z) := \text{Concat}(\mathbf{e}^{(1)}_{S_1}; \mathbf{e}^{(2)}_{S_2}; \ldots; \mathbf{e}^{(B)}_{S_B}) \in \mathbb{R}^d,$$

where

$$S_k = \{i \in [d_k] : |z^{(k)}_i| \textit{ ranks within the top-}\alpha\% \textit{ of all } |z^{(k)}|\textit{'s entries } (|z^{(k)}_1|, |z^{(k)}_2|, \ldots, |z^{(k)}_{d_k}|)\}$$

and $\mathbf{e}^{(k)}_{S_k}$ is a d_k-dimensional vector where the i-th entry is 1 if $i \in S_k$, and 0 otherwise.

Remark 1. *To ensure that the top-$\alpha\%$ filter $\texttt{FLT}_\alpha(z)$ is well-defined, when multiple entries share identical absolute values and including all of them in the set S_k would result in exceeding the $\alpha\%$ threshold of set size, the construction of S_k prioritizes the entries with the smallest indices among those with the same absolute values.*

Definition 2. *For any $z \in \mathbb{R}^d$, we define the $L_{1,\textit{top-}\alpha\%}$ norm of z as*

$$\|z\|_{1,\textit{top-}\alpha\%} := \|z \odot \texttt{FLT}_\alpha(z)\|_1.$$

Proposition 1. *$\|\cdot\|_{1,\textit{top-}\alpha\%}$ is indeed a norm in $\mathbb{R}^d$.*

Proof. By Definition 1, we get

$$\|z\|_{1,\text{top-}\alpha\%} = \|z \odot \texttt{FLT}_\alpha(z)\|_1 = \sum_{k=1}^{B} \|z^{(k)} \odot \mathbf{e}^{(k)}_{S_k}\|_1. \tag{5}$$

First, if $\|z\|_{1,\text{top-}\alpha\%} = 0$, then by (5), $\|z^{(k)} \odot \mathbf{e}^{(k)}_{S_k}\|_1 = 0$ for any $1 \leq k \leq B$. Thus,

$$\|z^{(k)}\|_\infty = \arg\max_{1 \leq i \leq d_k} |z^{(k)}_i| \leq \|z^{(k)} \odot \mathbf{e}^{(k)}_{S_k}\|_1 = 0.$$

So $z^{(k)}$ is a zero vector for any $1 \leq k \leq B$ and then z is a zero vector.

Second, for any given $c \in \mathbb{R}_+$, $\{|z^{(k)}_i|\}_{1 \leq i \leq d_k}$ and $\{|cz^{(k)}_i|\}_{1 \leq i \leq d_k}$ have the same order. So z and cz share the same filter $\texttt{FLT}_\alpha(z)$ and

$$\|cz\|_{1,\text{top-}\alpha\%} = \|cz \odot \texttt{FLT}_\alpha(cz)\|_1 = c\|z \odot \texttt{FLT}_\alpha(z)\|_1 = c\|z\|_{1,\text{top-}\alpha\%}.$$

Third, for any $x, y \in \mathbb{R}^d$, suppose that

$$\texttt{FLT}_\alpha(x) = \text{Concat}(\mathbf{e}^{(1)}_{S'_1}; \mathbf{e}^{(2)}_{S'_2}; \ldots; \mathbf{e}^{(B)}_{S'_B}) \quad \text{and} \quad \texttt{FLT}_\alpha(x+y) = \text{Concat}(\mathbf{e}^{(1)}_{S''_1}; \mathbf{e}^{(2)}_{S''_2}; \ldots; \mathbf{e}^{(B)}_{S''_B}).$$

By the construction of S'_k, for any $1 \leq k \leq B$, we have

$$\|x^{(k)} \odot \mathbf{e}^{(k)}_{S''_k}\|_1 \leq \|x^{(k)} \odot \mathbf{e}^{(k)}_{S'_k}\|_1.$$

So

$$\|x \odot \texttt{FLT}_\alpha(x+y)\|_1 = \sum_{k=1}^{B} \|x^{(k)} \odot \mathbf{e}^{(k)}_{S''_k} \leq \sum_{k=1}^{B} \|x^{(k)} \odot \mathbf{e}^{(k)}_{S'_k} = \|x \odot \texttt{FLT}_\alpha(x)\|_1.$$

Similarly, it holds that

$$\|y \odot \texttt{FLT}_\alpha(x+y)\|_1 \leq \|y \odot \texttt{FLT}_\alpha(y)\|_1.$$

Thus, we have

$$\begin{aligned}\|x+y\|_{1,\text{top-}\alpha\%} &= \|(x+y)\odot \mathtt{FLT}_\alpha(x+y)\|_1\\ &= \|x\odot \mathtt{FLT}_\alpha(x+y)+y\odot \mathtt{FLT}_\alpha(x+y)\|_1\\ &\le \|x\odot \mathtt{FLT}_\alpha(x+y)\|_1+\|y\odot \mathtt{FLT}_\alpha(x+y)\|_1\\ &\le \|x\odot \mathtt{FLT}_\alpha(x)\|_1+\|y\odot \mathtt{FLT}_\alpha(y)\|_1\\ &= \|x\|_{1,\text{top-}\alpha\%}+\|y\|_{1,\text{top-}\alpha\%}.\end{aligned}$$

□

We propose a lemma which is useful for the proof of Theorem 1.

Lemma 1. *For any $x, y \in \mathbb{R}^d$, it holds that*

$$\|x\odot FLT_\alpha(x)\|_1-\|x\odot FLT_\alpha(y)\|_1\le 2\|x-y\|_1.$$

Proof. By Proposition 1, $\|\cdot\|_{1,\text{top-}\alpha\%}$ is a norm in $\mathbb{R}^d$, so we have

$$\begin{aligned}&\|x\odot \mathtt{FLT}_\alpha(x)\|_1-\|x\odot \mathtt{FLT}_\alpha(y)\|_1\\ =\ &\|x\odot \mathtt{FLT}_\alpha(x)\|_1-\|y\odot \mathtt{FLT}_\alpha(y)\|_1+\|y\odot \mathtt{FLT}_\alpha(y)\|_1-\|x\odot \mathtt{FLT}_\alpha(y)\|_1\\ =\ &\|x\|_{1,\text{top-}\alpha\%}-\|y\|_{1,\text{top-}\alpha\%}+\|y\odot \mathtt{FLT}_\alpha(y)\|_1-\|x\odot \mathtt{FLT}_\alpha(y)\|_1\\ \le\ &\|x-y\|_{1,\text{top-}\alpha\%}+\|(y-x)\odot \mathtt{FLT}_\alpha(y)\|_1\\ \le\ &\|x-y\|_1+\|y-x\|_1\\ =\ &2\|x-y\|_1.\end{aligned}$$

□

C Proof of Theorem 1

Our proof of Theorem 1 follows the convergence analysis of the full-batch Adam optimizer in Shi et al. (2021), with novel adaptations to address the unique aspects of MoFO.

To maintain consistency with the notation used in MoFO (Algorithm 1 in Section D.4), we denote

$$z_t = \texttt{Concat}(z_t^{(1)}, \ldots, z_t^{(B)}),$$

where z represents the model parameter θ, the gradient g, the first moment estimate m, or the second moment estimate v. Notably, each of these variables belongs to $\mathbb{R}^d$. Thus, for any $1 \le i \le d$, we can denote $z_{i,t}$ as the i-th entry of z_t when z represents θ, g, m, or v.

By the update rules of the first and second moment estimates

$$m_{i,t} = (1-\beta_1) g_{i,t} + \beta_1 m_{i,t-1}, \quad m_{i,0} = 0,$$
$$v_{i,t} = (1-\beta_2) g_{i,t}^2 + \beta_2 v_{i,t-1}, \quad v_{i,0} = 0.$$

By mathematical induction, for any $1 \le i \le d$, we have

$$m_{i,t} = (1-\beta_1) \sum_{s=1}^{t} \beta_1^{t-s} g_{i,s} \tag{6}$$

and

$$v_{i,t} = (1-\beta_2) \sum_{s=1}^{t} \beta_2^{t-s} g_{i,s}^2. \tag{7}$$

We will frequently use Equation (6) and (7) in the proofs of the subsequent lemmas and theorems.

Lemma 2. *For the full-batch version of MoFO with hyperparameters satisfying $\beta_1 < \sqrt{\beta_2} < 1$, $\epsilon = 0$, it holds that*

$$|\theta_{i,t} - \theta_{i,t-1}| \le \frac{1}{\sqrt{1-\beta_2}(1-\beta_1/\sqrt{\beta_2})} \cdot \eta_t \cdot \texttt{FLT}_\alpha(m_t)_i, \quad \textit{for any coordinate } 1 \le i \le d.$$

Moreover, it holds that

$$\|\theta_t - \theta_{t-1}\|_2 \le C\eta_t,$$

where $C = \frac{\sqrt{d\cdot(\alpha\%)+B}}{\sqrt{1-\beta_2}(1-\beta_1/\sqrt{\beta_2})}$.

Proof. When the i-th entry is not in our filter at iteration t, i.e. $\texttt{FLT}_\alpha(m_t)_i = 0$, we have $\theta_{i,t} = \theta_{i,t-1}$. Then

$$|\theta_{i,t} - \theta_{i,t-1}| = 0 = \frac{1}{\sqrt{1-\beta_2}(1-\beta_1/\sqrt{\beta_2})} \cdot \eta_t \cdot \texttt{FLT}_\alpha(m_t)_i.$$

When the i-th entry is in our filter, i.e. $\texttt{FLT}_\alpha(m_t)_i = 1$, by the weight updating rule of MoFO, we have $\theta_{i,t} - \theta_{i,t-1} = -\eta_t \hat{m}_{i,t}/\sqrt{\hat{v}_{i,t}}$. We first analyze $m_{i,t}$ and $v_{i,t}$.

By Equation (6) and (7), we get

$$|m_{i,t}| \le (1-\beta_1) \sum_{s=1}^{t} \beta_1^{t-s} |g_{i,s}|,$$
$$v_{i,t} = (1-\beta_2) \sum_{s=1}^{t} \beta_2^{t-s} g_{i,s}^2 \ge (1-\beta_2)\beta_2^{t-s} g_{i,s}^2, \quad \text{for any } 1 \le s \le t.$$

So we get

$$
\begin{aligned}
|\theta_{i,t} - \theta_{i,t-1}| &= \left| -\eta_t \frac{\hat{m}_{i,t}}{\sqrt{\hat{v}_{i,t}}} \right| = \eta_t \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t} |m_{i,t}| / \sqrt{v_{i,t}} \\
&\leq \eta_t \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t} \sum_{s=1}^{t} \frac{(1-\beta_1)\beta_1^{t-s}|g_{i,s}|}{\sqrt{(1-\beta_2)\beta_2^{t-s}}|g_{i,s}|} = \eta_t \frac{1-\beta_1}{1-\beta_1^t} \sqrt{\frac{1-\beta_2^t}{1-\beta_2}} \sum_{s=1}^{t} (\beta_1/\sqrt{\beta_2})^{t-s} \\
&\leq \frac{\eta_t}{\sqrt{1-\beta_2}} \sum_{s=0}^{t-1} (\beta_1/\sqrt{\beta_2})^s \\
&\leq \frac{\eta_t}{\sqrt{1-\beta_2}(1-\beta_1/\sqrt{\beta_2})}.
\end{aligned}
$$

Here, the last inequality holds because of the assumption $\beta_1 < \sqrt{\beta_2} < 1$.

MoFO actually choose $\lceil d_k \times \alpha\% \rceil$ entries to update in each part k of parameters. Then for any $z \in \mathbb{R}^d$, we have

$$
\#\{1 \leq i \leq d : \mathtt{FLT}_\alpha(z)_i = 1\} = \sum_{k=1}^{B} \lceil d_k \cdot (\alpha\%) \rceil \leq \sum_{k=1}^{B} (d_k \cdot (\alpha\%) + 1) = d \cdot (\alpha\%) + B.
$$

Then for the L_2-distance, we have

$$
\begin{aligned}
\|\theta_t - \theta_{t-1}\|_2 &= \left(\sum_{k=1}^{d} |\theta_{i,t} - \theta_{i,t-1}|^2 \cdot \mathtt{FLT}_\alpha(m_t)_i \right)^{\frac{1}{2}} \\
&\leq \left(\frac{\eta_t^2}{(\sqrt{1-\beta_2}(1-\beta_1/\sqrt{\beta_2}))^2} \cdot \#\{1 \leq i \leq d : \mathtt{FLT}_\alpha(z)_i = 1\} \right)^{\frac{1}{2}} \\
&\leq \frac{\sqrt{d \cdot (\alpha\%) + B}}{\sqrt{1-\beta_2}(1-\beta_1/\sqrt{\beta_2})} \cdot \eta_t \\
&= C\eta_t.
\end{aligned}
$$

□

Lemma 3. *Suppose that the gradient $\nabla\mathcal{L}$ is Lipschitz continuous with constant L. Suppose that the full-batch version of MoFO has the hyperparameters satisfying $\beta_1 < \sqrt{\beta_2} < 1$, $\epsilon = 0$ and the learning rate schedule $\eta_t = \eta/\sqrt{t}$. For any iteration steps $t \geq s \geq 1$ and any coordinate i, it holds that*

$$
|g_{i,t} - g_{i,s}| \leq \|g_t - g_s\|_2 \leq \frac{2\sqrt{2}LC\eta(t-s)}{\sqrt{t}}.
$$

Proof. Since $\nabla\mathcal{L}$ has Lipschitz constant L, we get

$$
|g_{i,t} - g_{i,s}| \leq \|g_t - g_s\|_2 = \|\nabla\mathcal{L}(\theta_{t-1}) - \nabla\mathcal{L}(\theta_{t-1})\|_2 \leq L\|\theta_{t-1} - \theta_{s-1}\|_2. \tag{8}
$$

By Lemma 2, for any $t > s \geq 1$, we have

$$\begin{aligned}
\|\theta_{t-1} - \theta_{s-1}\|_2 &\leq \sum_{u=s}^{t-1} \|\theta_u - \theta_{u-1}\|_2 \leq C \sum_{u=s}^{t-1} \eta_u \\
&\leq C\eta \sum_{u=s}^{t-1} \frac{1}{\sqrt{u}} \leq C\eta \sum_{u=s}^{t-1} \frac{2}{\sqrt{u-1}+\sqrt{u}} \leq 2C\eta \sum_{u=s}^{t-1} (\sqrt{u} - \sqrt{u-1}) \\
&= 2C\eta(\sqrt{t-1} - \sqrt{s-1}) = \frac{2C\eta(t-s)}{\sqrt{t-1}+\sqrt{s-1}} \\
&\leq \frac{2C\eta(t-s)}{\sqrt{t-1}} \leq \frac{2C\eta(t-s)}{\sqrt{t/2}} \\
&= \frac{2\sqrt{2}C\eta(t-s)}{\sqrt{t}}.
\end{aligned}$$

When $t = s > 1$, it is obvious that

$$\|\theta_{t-1} - \theta_{s-1}\|_2 = 0 \leq \frac{2\sqrt{2}C\eta(t-s)}{\sqrt{t}}.$$

Combining it with (8), for any $t \geq s \geq 1$, we have

$$|g_{i,t} - g_{i,s}| \leq \|g_t - g_s\|_2 \leq \frac{2\sqrt{2}LC\eta(t-s)}{\sqrt{t}}.$$

$\square$

Lemma 4. *Under the assumptions in Lemma 3, for any iteration step $t \geq 1$ and any coordinate i, it holds that*

$$g_{i,t}\frac{\hat{m}_{i,t}}{\sqrt{\hat{v}_{i,t}}} \geq \sqrt{1-\beta_2}\left(|g_{i,t}| - \left[\frac{2\sqrt{2}\beta_1}{(1-\beta_1)^2} + \frac{4}{1-\beta_2}\right]\frac{LC\eta}{\sqrt{t}}\right).$$

Proof. By Lemma 3, we get

$$g_{i,t}g_{i,s} = g_{i,t}^2 - g_{i,t}(g_{i,t} - g_{i,s}) \geq g_{i,t}^2 - |g_{i,t}| \cdot |g_{i,t} - g_{i,s}| \geq g_{i,t}^2 - \frac{2\sqrt{2}LC\eta(t-s)}{\sqrt{t}}|g_{i,t}|.$$

Then we have

$$\begin{aligned}
g_{i,t}m_{i,t} &= (1-\beta_1)\sum_{s=1}^{t} \beta_1^{t-s} g_{i,t}g_{i,s} \\
&\geq g_{i,t}^2 \cdot (1-\beta_1)\sum_{s=1}^{t}\beta_1^{t-s} - \frac{2\sqrt{2}LC\eta}{\sqrt{t}}|g_{i,t}| \cdot (1-\beta_1)\sum_{s=1}^{t}\beta_1^{t-s}\cdot(t-s) \qquad (9) \\
&\geq g_{i,t}^2 \cdot (1-\beta_1)\sum_{s=0}^{t-1}\beta_1^{s} - \frac{2\sqrt{2}LC\eta}{\sqrt{t}}|g_{i,t}| \cdot (1-\beta_1)\sum_{s=1}^{t-1} s\beta_1^{s}.
\end{aligned}$$

Since we have

$$\sum_{s=0}^{t-1}\beta_1^s = \frac{1-\beta_1^t}{1-\beta_1}, \quad \sum_{s=1}^{t-1} s\beta_1^{s-1} \leq \sum_{s=1}^{\infty} s\beta_1^{s-1} = \frac{d}{d\beta_1}\left(\sum_{s=1}^{\infty}\beta_1^s\right) = \frac{d}{d\beta_1}\left(\frac{\beta_1}{1-\beta_1}\right) = \frac{1}{(1-\beta_1)^2}, \qquad (10)$$

it holds that

$$g_{i,t}m_{i,t} \geq \text{RHS of (9)} \geq (1-\beta_1^t)g_{i,t}^2 - \frac{2\sqrt{2}\beta_1 LC\eta}{(1-\beta_1)\sqrt{t}}|g_{i,t}|. \qquad (11)$$

For the second moment estimate, we have

$$\begin{aligned}
v_{i,t} &= (1-\beta_2)\sum_{s=1}^{t}\beta_2^{t-s}g_{i,s}^2 \le (1-\beta_2)\sum_{s=1}^{t}\beta_2^{t-s}(|g_{i,t}|+|g_{i,s}-g_{i,t}|)^2 \\
&\le (1-\beta_2)\sum_{s=1}^{t}\beta_2^{t-s}\left(|g_{i,t}|+\frac{2\sqrt{2}LC\eta(t-s)}{\sqrt{t}}\right)^2 = (1-\beta_2)\sum_{s=0}^{t-1}\beta_2^{s}\left(|g_{i,t}|+\frac{2\sqrt{2}LC\eta s}{\sqrt{t}}\right)^2 \\
&= |g_{i,t}|^2\cdot(1-\beta_2)\left(\sum_{s=0}^{t-1}\beta_2^s\right) + |g_{i,t}|\cdot\frac{4\sqrt{2}LC\eta}{\sqrt{t}}(1-\beta_2)\left(\sum_{s=1}^{t-1}s\beta_2^s\right) \\
&\quad + \frac{8L^2C^2\eta^2}{t}(1-\beta_2)\left(\sum_{s=1}^{t-1}s^2\beta_2^s\right).
\end{aligned} \tag{12}$$

Since we have

$$\begin{aligned}
\sum_{s=0}^{t-1}\beta_2^s &= \frac{1-\beta_2^t}{1-\beta_2} \le \frac{1}{1-\beta_2}, \\
\sum_{s=0}^{t-1}s\beta_2^{s-1} &\le \sum_{s=0}^{\infty}s\beta_2^{s-1} = \frac{d}{d\beta_2}\left(\sum_{s=0}^{\infty}\beta_2^s\right) = \frac{d}{d\beta_2}\left(\frac{1}{1-\beta_2}\right) = \frac{1}{(1-\beta_2)^2}, \\
\sum_{s=0}^{t-1}s^2\beta_2^{s-1} &\le \sum_{s=0}^{\infty}s^2\beta_2^{s-1} = \beta_2\left(\sum_{s=0}^{\infty}s(s-1)\beta_2^{s-2}\right) + \sum_{s=0}^{\infty}s\beta_2^{s-1} \\
&= \beta_2\cdot\frac{d^2}{d\beta_2^2}\left(\sum_{s=0}^{\infty}\beta_2^s\right) + \frac{1}{(1-\beta_2)^2} = \beta_2\cdot\frac{d^2}{d\beta_2^2}\left(\frac{1}{1-\beta_2}\right) + \frac{1}{(1-\beta_2)^2} \\
&= \frac{2\beta_2}{(1-\beta_2)^3} + \frac{1}{(1-\beta_2)^2} \\
&= \frac{1+\beta_2}{(1-\beta_2)^3},
\end{aligned}$$

it holds that

$$\begin{aligned}
v_{i,t} \le \text{RHS of (12)} &\le |g_{i,t}|^2 + |g_{i,t}|\cdot\frac{4\sqrt{2}\beta_2 LC\eta}{(1-\beta_2)\sqrt{t}} + \frac{8(1+\beta_2)\beta_2L^2C^2\eta^2}{(1-\beta_2)^2t} \\
&\le |g_{i,t}|^2 + |g_{i,t}|\cdot\frac{8LC\eta}{(1-\beta_2)\sqrt{t}} + \frac{16L^2C^2\eta^2}{(1-\beta_2)^2t} \\
&= \left(|g_{i,t}| + \frac{4LC\eta}{(1-\beta_2)\sqrt{t}}\right)^2.
\end{aligned}$$

Thus, we get

$$\sqrt{v_{i,t}} \le |g_{i,t}| + \frac{4LC\eta}{(1-\beta_2)\sqrt{t}}.$$

Recalling (11), we have

$$\begin{aligned} g_{i,t}m_{i,t} &\geq (1-\beta_1^t)\left(|g_{i,t}| + \frac{4LC\eta}{(1-\beta_2)\sqrt{t}}\right)\left(|g_{i,t}| - \frac{2\sqrt{2}\beta_1 LC\eta}{(1-\beta_1^t)(1-\beta_1)\sqrt{t}} - \frac{4LC\eta}{(1-\beta_2)\sqrt{t}}\right) \\ &\quad + (1-\beta_1^t)\cdot\frac{4LC\eta}{(1-\beta_2)\sqrt{t}}\left(\frac{2\sqrt{2}\beta_1 LC\eta}{(1-\beta_1^t)(1-\beta_1)\sqrt{t}} + \frac{4LC\eta}{(1-\beta_2)\sqrt{t}}\right) \\ &\geq (1-\beta_1^t)\left(|g_{i,t}| + \frac{4LC\eta}{(1-\beta_2)\sqrt{t}}\right)\left(|g_{i,t}| - \frac{2\sqrt{2}\beta_1 LC\eta}{(1-\beta_1^t)(1-\beta_1)\sqrt{t}} - \frac{4LC\eta}{(1-\beta_2)\sqrt{t}}\right) \\ &\geq (1-\beta_1^t)\sqrt{v_{i,t}}\left(|g_{i,t}| - \frac{2\sqrt{2}\beta_1 LC\eta}{(1-\beta_1^t)(1-\beta_1)\sqrt{t}} - \frac{4LC\eta}{(1-\beta_2)\sqrt{t}}\right). \end{aligned}$$

Therefore,

$$\begin{aligned} g_{i,t}\frac{\hat{m}_{i,t}}{\sqrt{\hat{v}_{i,t}}} = \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t}g_{i,t}\frac{m_{i,t}}{\sqrt{v_{i,t}}} &\geq \sqrt{1-\beta_2^t}\left(|g_{i,t}| - \frac{2\sqrt{2}\beta_1 LC\eta}{(1-\beta_1^t)(1-\beta_1)\sqrt{t}} - \frac{4LC\eta}{(1-\beta_2)\sqrt{t}}\right) \\ &\geq \sqrt{1-\beta_2}\left(|g_{i,t}| - \left[\frac{2\sqrt{2}\beta_1}{(1-\beta_1)^2} + \frac{4}{1-\beta_2}\right]\frac{LC\eta}{\sqrt{t}}\right). \end{aligned}$$

□

Lemma 5. *Under the assumptions in Lemma 3, for any iteration step $t \geq 1$ and any coordinate i, it holds that*

$$\left\|\frac{m_t}{1-\beta_1^t} - g_t\right\|_1 \leq \frac{2\sqrt{2}\beta_1\sqrt{d}LC\eta}{(1-\beta_1)^2\sqrt{t}}.$$

Proof. Recalling (6), we get

$$m_t = (1-\beta_1)\sum_{s=1}^{t}\beta_1^{t-s}g_s,$$

and

$$m_t - (1-\beta_1^t)g_t = (1-\beta_1)\sum_{s=1}^{t}\beta_1^{t-s}(g_t - g_s).$$

By Lemma 3 and Equation (10) in the proof of Lemma 4, we get

$$\begin{aligned} \left\|\frac{m_t}{1-\beta_1^t} - g_t\right\|_2 &\leq \frac{1-\beta_1}{1-\beta_1^t}\sum_{s=1}^{t}\beta_1^{t-s}\|g_t - g_s\|_2 \leq \sum_{s=1}^{t}\beta_1^{t-s}\|g_t - g_s\|_2 \\ &\leq \frac{2\sqrt{2}LC\eta}{\sqrt{t}}\sum_{s=1}^{t}\beta_1^{t-s}(t-s) = \frac{2\sqrt{2}LC\eta}{\sqrt{t}}\sum_{s=0}^{t-1}s\beta_1^s \\ &\leq \frac{2\sqrt{2}\beta_1 LC\eta}{(1-\beta_1)^2\sqrt{t}}. \end{aligned}$$

By Cauchy-Schwarz's inequality, we have

$$\left\|\frac{m_t}{1-\beta_1^t} - g_t\right\|_1 \leq \sqrt{d}\left\|\frac{m_t}{1-\beta_1^t} - g_t\right\|_2 \leq \frac{2\sqrt{2}\beta_1\sqrt{d}LC\eta}{(1-\beta_1)^2\sqrt{t}}.$$

□

Now we will complete the proof of Theorem 1.

Proof of Theorem 1. By the descent lemma, since $\nabla\mathcal{L}$ is Lipschitz with constant L, we have

$$\begin{aligned}\mathcal{L}(\theta_t)-\mathcal{L}(\theta_{t-1}) &\le \nabla\mathcal{L}(\theta_{t-1})^\top(\theta_t-\theta_{t-1})+\frac{L}{2}\|\theta_t-\theta_{t-1}\|_2^2\\ &\le g_t^\top(\theta_t-\theta_{t-1})+\frac{L}{2}\|\theta_t-\theta_{t-1}\|_2^2.\end{aligned}\tag{13}$$

By Lemma 2 and Lemma 4, we have

$$\begin{aligned}&\mathcal{L}(\theta_t)-\mathcal{L}(\theta_{t-1}) \le \text{RHS of (13)} \le -\eta_t\left(\sum_{i=1}^{d} g_{i,t}\frac{\hat{m}_{i,t}}{\sqrt{\hat{v}_{i,t}}}\cdot\mathtt{FLT}_\alpha(m_t)_i\right)+\frac{LC^2\eta_t^2}{2}\\ &\le \frac{LC^2\eta^2}{2t}-\frac{\eta}{\sqrt{t}}\sum_{i=1}^{d}\sqrt{1-\beta_2}\left(|g_{i,t}|-\left[\frac{2\sqrt{2}\beta_1}{(1-\beta_1)^2}+\frac{4}{1-\beta_2}\right]\frac{LC\eta}{\sqrt{t}}\right)\cdot\mathtt{FLT}_\alpha(m_t)_i\\ &= -\frac{\sqrt{1-\beta_2}\cdot\eta}{\sqrt{t}}\|g_t\odot\mathtt{FLT}_\alpha(m_t)\|_1+\left[\frac{2\sqrt{2}\beta_1\sqrt{1-\beta_2}}{(1-\beta_1)^2}+\frac{4}{\sqrt{1-\beta_2}}+\frac{C}{2}\right]\frac{LC\eta^2}{t}\cdot\|\mathtt{FLT}_\alpha(m_t)\|_1\\ &\le -\frac{\sqrt{1-\beta_2}\cdot\eta}{\sqrt{t}}\|g_t\odot\mathtt{FLT}_\alpha(m_t)\|_1+\left[\frac{2\sqrt{2}\beta_1\sqrt{1-\beta_2}}{(1-\beta_1)^2}+\frac{4}{\sqrt{1-\beta_2}}+\frac{C}{2}\right]\frac{LC\eta^2(d\cdot(\alpha\%)+B)}{t}.\end{aligned}\tag{14}$$

By Lemma 1 and Lemma 5, we have

$$\begin{aligned}\|g_t\odot\mathtt{FLT}_\alpha(g_t)\|_1-\|g_t\odot\mathtt{FLT}_\alpha(m_t)\|_1 &= \|g_t\odot\mathtt{FLT}_\alpha(g_t)\|_1-\left\|g_t\odot\mathtt{FLT}_\alpha\left(\frac{m_t}{1-\beta_1^t}\right)\right\|_1\\ &\le 2\left\|g_t-\frac{m_t}{1-\beta_1^t}\right\|_1\\ &\le \frac{4\sqrt{2}\beta_1\sqrt{d}LC\eta}{(1-\beta_2)^2\sqrt{t}}.\end{aligned}$$

Thus,

$$\begin{aligned}&\mathcal{L}(\theta_t)-\mathcal{L}(\theta_{t-1}) \le \text{RHS of (14)}\\ &\le -\frac{\sqrt{1-\beta_2}\cdot\eta}{\sqrt{t}}\|g_t\odot\mathtt{FLT}_\alpha(g_t)\|_1+\left[\frac{2\sqrt{2}\beta_1\sqrt{1-\beta_2}}{(1-\beta_1)^2}+\frac{4}{\sqrt{1-\beta_2}}+\frac{C}{2}\right]\frac{LC\eta^2(d\cdot(\alpha\%)+B)}{t}\\ &\qquad\qquad+\frac{4\sqrt{2}\beta_1\sqrt{d}LC\eta^2}{(1-\beta_2)^{\frac{3}{2}}t}\\ &= -\frac{C_1}{\sqrt{t}}\|g_t\|_{1,\text{top-}\alpha\%}+\frac{C_2}{t}\le -\frac{C_1}{\sqrt{t}}\min_{1\le t\le T}\|g_t\|_{1,\text{top-}\alpha\%}+\frac{C_2}{t},\end{aligned}\tag{15}$$

where

$$\begin{aligned}C_1 &= \sqrt{1-\beta_2}\cdot\eta,\\ C_2 &= LC\eta^2\cdot\left\{\left[\frac{2\sqrt{2}\beta_1\sqrt{1-\beta_2}}{(1-\beta_1)^2}+\frac{4}{\sqrt{1-\beta_2}}+\frac{C}{2}\right](d\cdot(\alpha\%)+B)+\frac{4\sqrt{2}\beta_1\sqrt{d}}{(1-\beta_2)^{\frac{3}{2}}}\right\}.\end{aligned}$$

Taking the summation of (14) from 1 to T, we get

$$\begin{aligned}\mathcal{L}^*-\mathcal{L}(\theta_0)\le\mathcal{L}(\theta_T)-\mathcal{L}(\theta_0) &= \sum_{t=1}^{T}\mathcal{L}(\theta_t)-\mathcal{L}(\theta_{t-1})\\ &\le -C_1\left(\sum_{t=1}^{T}\frac{1}{\sqrt{t}}\right)\cdot\min_{1\le t\le T}\|g_t\odot\mathtt{FLT}_\alpha(g_t)\|_1+C_2\sum_{t=1}^{T}\frac{1}{t}.\end{aligned}$$

Since

$$\sum_{t=1}^{T} \frac{1}{\sqrt{t}} \geq \sum_{t=1}^{T} \frac{2}{\sqrt{t}+\sqrt{t+1}} = \sum_{t=1}^{T} 2(\sqrt{t+1}-\sqrt{t}) = 2(\sqrt{T+1}-1),$$

$$\sum_{t=1}^{T} \frac{1}{t} = 1 + \sum_{t=1}^{T-1} \frac{1}{t+1} \leq 1 + \sum_{t=1}^{T-1} \int_{t}^{t+1} \frac{1}{u}\, du \leq 1 + \int_{1}^{T} \frac{1}{u}\, du = 1 + \log T,$$

we get

$$\begin{aligned}
\min_{0 \leq t \leq T-1} \|\nabla \mathcal{L}(\theta_t)\|_\infty = \min_{1 \leq t \leq T} \|g_t\|_\infty &\leq \min_{1 \leq t \leq T} \|g_t\|_{1,\text{top-}\alpha\%} \\
&\leq \frac{\mathcal{L}(\theta_0) - \mathcal{L}^* + C_2 \sum_{t=1}^{T} \frac{1}{t}}{C_1 \sum_{t=1}^{T} \frac{1}{\sqrt{t}}} \leq \frac{\mathcal{L}(\theta_0) - \mathcal{L}^* + C_2(1+\log T)}{2C_1(\sqrt{T+1}-1)} \\
&= \mathcal{O}\left(\frac{\log T}{\sqrt{T}}\right).
\end{aligned}$$

□

D Implementation Details

D.1 Datasets for Fine-Tuning.

MetaMathQA (Yu et al., 2024b). This dataset comprises 395K math question-answer pairs. Numerous studies indicate that LLMs significantly enhance performance metrics on mathematical benchmarks such as GSM8K after fine-tuning on this dataset. We randomly select 10% of this dataset for training LLMs, which includes 39.5K question-answer pairs.

PMC-LLaMA-Instructions (Wu et al., 2024). This dataset comprises 514K instruction-response pairs. Fine-tuning LLMs on this dataset has been shown to enhance performance on medical NLP tasks, such as PubMedQA (Jin et al., 2019), MedMCQA (Pal et al., 2022), and MedQA (Jin et al., 2021). We randomly sampled 51K instances with prompt lengths less than 750 characters for training our models.

TRACE benchmark dataset (Wang et al., 2023b). TRACE benchmark is designed with a comprehensive set of 8 distinct tasks across various domains, including domain-specific knowledge, multilingual proficiency, code generation, and mathematical reasoning.

D.2 Evaluation Metrics for Instruction Fine-Tuning

We employ a comprehensive suite of widely used benchmarks to assess the performance and potential catastrophic forgetting effects on the general capabilities of LLMs after instruction fine-tuning. The benchmarks are as follows:

- **Factual knowledge (MMLU)**: We use the Massive Multitask Language Understanding (MMLU) benchmark (Hendrycks et al., 2021) to evaluate factual knowledge across 57 diverse subjects, ranging from STEM fields and the humanities to social sciences. Evaluations are performed using 8-bit precision with the open-instruct implementation, and by following the setup of (Hui et al., 2024), we report the 0-shot accuracy.
- **Common sense reasoning (CommonSense)**: To measure the commonsense reasoning capabilities of LLMs, we employ the widely recognized benchmarks ARC-Challenge, ARC-Easy (Clark et al., 2018), and HellaSwag (Zellers et al., 2019), collectively referred to as the Commonsense benchmark. We use the average of their metrics as the evaluation, conducting assessments using the LM Eval Harness framework (Gao et al., 2023) and reporting the 0-shot accuracy based on the "acc_norm, none" metric.
- **Mathematical Reasoning (GSM8K)**: We assess mathematical reasoning capability using GSM8K (Cobbe et al., 2021), which consists of 8.5K high-quality grade school math problems. Evaluations are conducted on the test set using the LM Eval Harness framework prompting in a 5-shot setting, reporting the "exact_match, flexible-extract" metric.
- **Code Generation (HumanEval)**: We adopt HumanEval (Chen et al., 2021), comprising 164 unique programming problems, to evaluate the coding capabilities of LLMs. For chat experiments, we use the vLLM framework with the open-instruct implementation and report the pass@10 performance.
- **Medical Question Answering (MedQ)**: To assess medical knowledge, we utilize three benchmarks—PubMedQA (Jin et al., 2019), MedMCQA (Pal et al., 2022), and MedQA (Jin et al., 2021). Evaluations are performed using the LM Eval Harness framework. For PubMedQA, we report the "acc, none" metric; for MedMCQA and MedQA, we report the "acc_norm, none" metric.
- **Instruction Following (IFEval)**: We evaluate the instruction-following ability of LLMs using the IFeval benchmark. Evaluations are conducted with the LM Eval Harness implementation, and we report the "inst_level_strict_acc, none" metric.

All benchmarks—including CommonSense, GSM8K, PubMedQA, MedMCQA, MedQA, and IFeval—are evaluated using the LM Eval Harness framework (Gao et al., 2023), following their default settings unless specified otherwise.

D.3 Hyperparameter Configurations

Instruction fine-tuning. In our instruction fine-tuning experiments, we follow the implementation of Ivison et al. (2023). For instruction fine-tuning, we set the maximum sequence length to 1024, the global batch size to 128, and we train the model for 2 epochs. For the Llama-2-7B model, we use a learning rate of 2e-5, with a cosine decay learning rate scheduler. The learning rate is set to 2e-5 for fine-tuning both the Llama-2-7B-Chat model on the MetaMathQA dataset and the Gemma-2B-IT model, while a learning rate of 1e-5 is used for fine-tuning the Llama-2-7B-Chat model on the PMC-LLaMA-Instruct dataset; all these settings employ a warm-up ratio of 0.03 and a cosine decay learning rate scheduler. For LoRA, we set the learning rate as 1e-4. The other hyperparameters in the experiments are as follows.

Fine-tuning Llama-2-7B on MetaMathQA.

- Learning rate: 2e-5.
- Update fraction of MoFO: $\alpha\% = 15\%$.
- LoRA: $r = 4, 16, 64, 256$. We report the best-performing hyperparameter configuration for the fine-tuning task in Table 1, which, in this case, is $r = 256$.

Fine-tuning Llama-2-7B-Chat on PMC-LLaMA-Instruct.

- Learning rate: 1e-5.
- Update fraction of MoFO: $\alpha\% = 10\%$.
- LoRA: $r = 16, 256$. We report the best-performing hyperparameter configuration for the fine-tuning task in Table 5, which, in this case, is $r = 256$.

Fine-tuning Llama-2-7B-Chat on MetaMathQA.

- Learning rate: 2e-5.
- Update fraction of MoFO: $\alpha\% = 15\%$.
- LoRA: $r = 16, 256$. We report the best-performing hyperparameter configuration for the fine-tuning task in Table 7, which, in this case, is $r = 256$.

Fine-tuning Gemma-2B-IT on MetaMathQA.

- Learning rate: 2e-5.
- Update fraction of MoFO: $\alpha\% = 5\%$.
- LoRA: $r = 16, 256, 512$. We report the best-performing hyperparameter configuration for the fine-tuning task in Table 6, which, in this case, is $r = 512$.

Hyperparameters in the Pareto comparison. To provide a comprehensive comparison, we explore various hyperparameter settings for λ_1, λ_2, LoRA's rank, and the update fraction $\alpha\%$ in MoFO in Figure 4. Specifically, we set λ_1 as 1e-4, 1e-5, 1e-6, 1e-7, while λ_2 is set as 1e-2, 5e-3, 1e-3, 5e-4, and 1e-4. The update fraction $\alpha\%$ in MoFO is set as 5%, 10%, 15%, 20%, 40%, 80%. The rank of LoRA is set as 4, 16, 64, 256.

Continual fine-tuning. In our continual fine-tuning experiments, we follow the default settings of the TRACE benchmark. We sequentially train TinyLlama-1.1B on the TRACE benchmark datasets: C-STANCE, FOMC, MeetingBank, Py150, ScienceQA, NumGLUE-cm, NumGLUE-ds, and 20Minuten for 5, 3, 7, 5, 3, 5, 5, and 7 epochs, respectively. We use a learning rate of 1e-5 with a cosine decay schedule and a batch size of 64. The parameter update fraction for MoFO is set to 5%.

All experiments are conducted on four A800 (80GB) GPUs.

D.4 More Explanation on the partitioning and Calculation of distance

Partitioning. We use the default partitioning scheme in PyTorch's Transformer implementation. Different types of parameters within the Transformer, such as query (Q), key (K), value (V) weights for attention heads, and feed-forward network (FFN) weights, are divided into separate partitions. Notably, in the default PyTorch implementation, within a layer, the query (Q) weights of all attention heads are grouped into a single partition. The same applies to the key (K) and value (V) weights. Our momentum-based filtering mechanism is applied to each partition individually.

Calculation of distance. Following the notation in Section , we suppose that the parameter parameters are partitioned into

$$\theta = (\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(B)}).$$

Denote the pre-trained model by θ_0 and the fine-tuned model by θ.

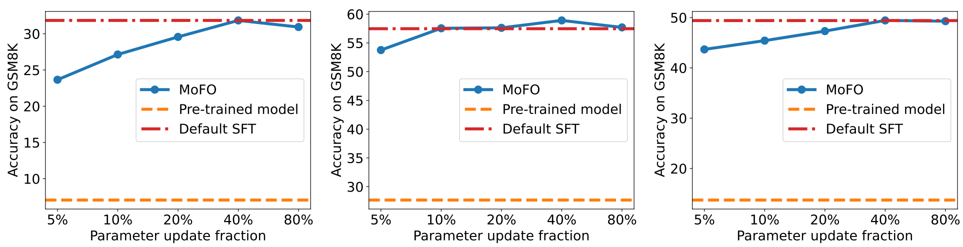
First, we calculate the relative change of parameters $\frac{\|\theta^{(k)} - \theta_0^{(k)}\|}{\|\theta_0^{(k)}\|}$ in each partition $k \in \{1, 2, \ldots, B\}$. Second, we compute the distance from the pre-trained model θ_0 to the fine-tuned model θ by averaging the relative changes across all partitions, defined as:

$$D(\theta, \theta_0) = \frac{1}{B} \sum_{k=1}^{B} \frac{\|\theta^{(k)} - \theta_0^{(k)}\|}{\|\theta_0^{(k)}\|}.$$

E Additional Experiments

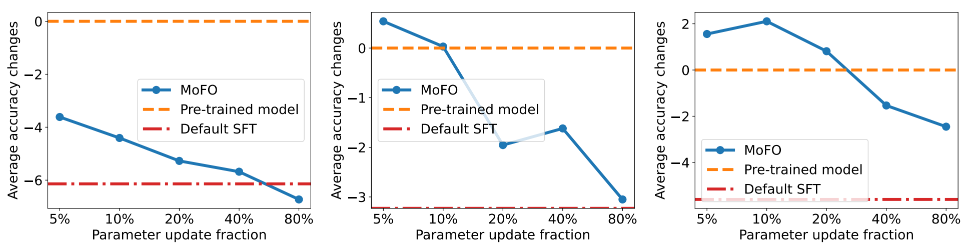
E.1 Impact of the Update Fraction

In this section, we first investigate the impact of the update fraction of parameters in the MoFO algorithm at each iteration, and then explore the effects of different update strategies within MoFO.



(a) Llama-3.2-1B (b) Llama-3.2-3B (c) Llama-2-7B

Figure 6: The performance of LLMs with different sizes on the math reasoning task (GSM8K) after fine-tuning on MetaMathQA using MoFO with different update fractions ($\alpha\%$) of parameters. Results show that across models of different sizes, setting the fraction $\alpha\%$ to approximately 20% allows MoFO to reach fine-tuning performance similar to the default FT (with up to 3% performance drop).



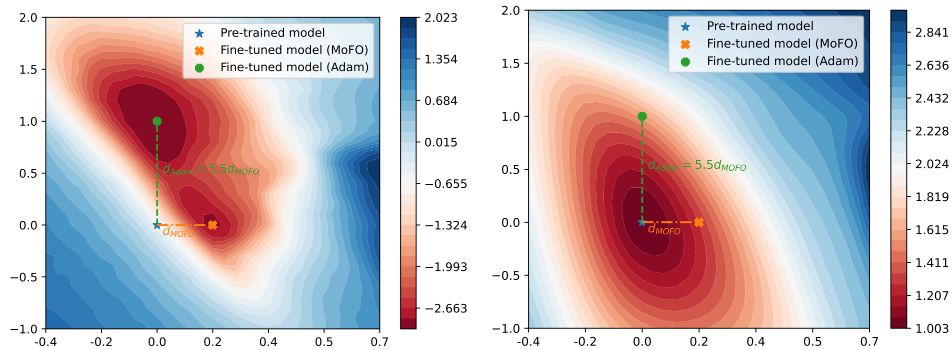
(a) Llama-3.2-1B (b) Llama-3.2-3B (c) Llama-2-7B

Figure 7: Average accuracy changes on MMLU, HumanEval, Commonsense Reasoning benchmarks compared to the pre-trained LLMs of different sizes after fine-tuning on MetaMathQA using MoFO with different update fractions ($\alpha\%$) of parameters. Larger LLMs tend to retain their pre-training knowledge more effectively when fine-tuned with MoFO, even when using smaller fractions of parameter updates.

Impact of update fraction of parameters in MoFO. Following the setting in Section 4.2, we fine-tune Llama-3.2-1B, Llama-3.2-3B, and Llama-2-7B on the MetaMathQA dataset using MoFO with varying update fractions of parameters at each iteration for 2 epochs. The experimental results of math reasoning (GSM8K) and average general capability performance changes are presented in Figure 6 and Figure 7.

The parameter update fraction affects the fine-tuning performance. Figure 6 shows that larger update fractions can improve MoFO's optimization effectiveness. Furthermore, in Llama-2-7B and Llama-3.2-3B, MoFO with a 5% parameter update fraction is sufficient to achieve nearly 90% of the performance of Default FT. Besides, experimental results show that setting the update fraction as α to approximately 20% enables MoFO to attain fine-tuning performance comparable to the default FT across various model sizes.

The parameter update fraction also affects the preservation of general capabilities. Figure 7 indicates that larger LLMs effectively maintain their pre-training knowledge when fine-tuned with MoFO, especially when using update fraction α less than 10%. Beyond the threshold of 20%, further increases in the parameter update fraction lead to a decline in general capabilities. Despite this, MoFO still forgets significantly less than Default FT in larger LLMs.



(a) Loss landscape on fine-tuning dataset

(b) Loss landscape on pre-training dataset

Figure 8: The loss landscapes of Pythia-160m after fine-tuning on a subset of the FLAN dataset using Adam optimizer and MoFO. We plot the loss landscapes on (a) the fine-tuning dataset and (b) the pre-training dataset (Pile). A logarithmic scale is applied to the loss values for better visualization. We find that MoFO, reaching a closer point to the pre-trained model, has minimal fine-tuning loss and lower pre-training loss, compared to Adam.

Table 4: Pythia-160m's performance on common sense tasks, after being fine-tuned with the Adam optimizer and MoFO. The results indicate that MoFO significantly mitigates catastrophic forgetting. Bold values denote the best results among these optimizers.

	HellaSwag	ARC-easy	ARC-challenge	Average
Pythia-160m	30.1	39.6	23.8	31.2
Adam	28.3	37.4	22.1	29.3
MoFO	**29.9**	**42.0**	**22.9**	**31.6**

In summary, MoFO can preserve pre-training knowledge and significantly enhance fine-tuning performance by choosing a moderate update fraction, avoiding the extremes of too small or too large fractions.

E.2 Validating MoFO's Impact on Preserving Pre-training Knowledge through Proximity

In this section, we empirically examine whether MoFO achieves its intended goal of converging to a minimum closer to the pre-trained model and mitigating forgetting mentioned in Section 3.

Our exploratory experiment shows that MoFO indeed converges to a minimum closer to the pre-training model. As shown in Figure 8(a), both MoFO and the Adam optimizer achieve minimal fine-tuning loss, indicating that switching from Adam to MoFO does not lead to performance degradation. Moreover, the distance from the pre-trained model to the minimum reached by MoFO is approximately 20% of that reached by the default Adam optimizer.

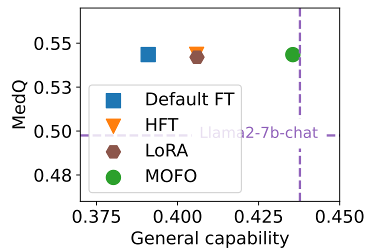
Our experiment demonstrates that the reduced parameter movement achieved by MoFO effectively mitigates the forgetting of pre-training knowledge. As shown in Figure 8(b), the fine-tuned model using MoFO experiences a smaller increase in pre-training loss. Additionally, Table 4 shows that MoFO achieves higher accuracy on commonsense reasoning tasks, indicating less forgetting.

E.3 More Experimental Results in Instruction Fine-Tuning

Results of fine-tuning on PMC-LLaMA-Instruct. We fine-tune Llama-2-7B-Chat on the PMC-LLaMA-Instructions dataset using various baseline methods and present the experimental results on medical question answering (MedQ) and general capabilities in Table 5. Since the MMLU benchmark

Table 5: The performance on the fine-tuning task (medical QA task), measured by MedQ, and general capability scores of Llama-2-7B-Chat after fine-tuning on the PMC-LLaMA-Instruct dataset. The figure on the right visualizes both MedQ accuracy and general capability scores. The results show that MoFO achieves comparable performance in the MedQ while significantly mitigating forgetting of general capabilities. Bold values denote the best results among these methods.

Method	MedQ	General Capability			
		CR	IFEval	HumanEval	Avg.
Llama-2-7B-Chat	49.8	65.6	41.4	24.3	43.8
Default FT	54.3	64.6	32.1	20.6	39.1
HFT	**54.4**	65.2	33.5	23.1	40.6
LoRA	54.2	64.4	33.9	23.5	40.6
MoFO	54.3	**65.5**	**41.1**	**24.1**	**43.6**

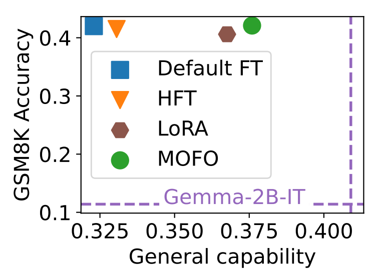


already contains medical-related instances (Hendrycks et al., 2021), which may lead to improved performance after fine-tuning, we instead use IFEval to assess general capabilities.

MoFO performs well on the fine-tuning task of medical QA. It achieves compatible performance compared to Default FT and HFT. In terms of general capabilities, MoFO demonstrates the least degradation compared to other baselines, with an average accuracy reduction of only 0.2%. Specifically, on the IFEval benchmark, our method only exhibits a minor reduction of 0.3%, while Default FT, HFT, and LoRA experience significant degradations ranging from 7.5% to 9.3%. On code generation (HumanEval) tasks and commonsense reasoning (CR) benchmarks, our method also only exhibits a minor reduction less than 0.2%.

Table 6: The performance of the fine-tuning task (math), measured by GSM8K, and the general capability scores of Gemma-2B-IT after fine-tuning on the MetaMathQA dataset. The figure on the right visualizes both GSM8K accuracy and general capability scores. The results show that MoFO achieves comparable performance in the fine-tuning task, while significantly mitigating forgetting of general capabilities. Bold values denote the best results among these methods.

Method	GSM8K	General Capability			
		CR	IFeval	HumanEval	Avg.
Gemma-2B-IT	11.4	57.6	33.6	31.5	40.9
Default FT	42.0	52.1	24.3	20.6	32.3
HFT	41.5	53.9	24.1	21.2	33.1
LoRA	40.6	54.4	26.1	**29.8**	36.8
MoFO	**42.1**	**55.0**	**28.7**	29.1	**37.6**



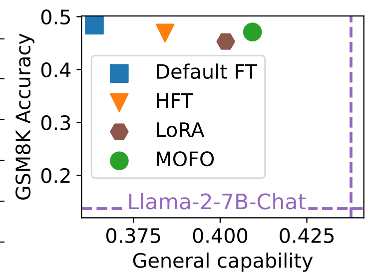
Results of Gemma-2B-IT fine-tuning on MetaMathQA. We also explore how MoFO performs in other LLMs. Specifically, we fine-tune Gemma-2B-IT on MetaMathQA using various baseline methods and present the experimental results on mathematical reasoning (GSM8K) and general capabilities in Table 6. The experimental results demonstrate that MoFO achieves comparable performance of the fine-tuning task to Default FT and HFT across different models. In terms of general capabilities, MoFO exhibits significantly less forgetting compared to other baselines. This result demonstrates the versatility of the MoFO algorithm.

We also fine-tune the Llama-2-7B-Chat on the MetaMathQA dataset. The results are presented in Table 7. The results demonstrate that our approach achieves performance comparable to Default FT and HFT while exhibiting less forgetting compared to baseline methods.

In summary, our MoFO algorithm shows competitive performance in instruction fine-tuning while preserving the general capabilities, effectively alleviating forgetting.

Table 7: The performance of the fine-tuning task (math), measured by GSM8K, and the general capability scores of Llama-2-7B-chat after fine-tuning on the MetaMathQA dataset. The figure on the right visualizes both GSM8K accuracy and general capability scores. The results show that MoFO achieves comparable performance in the fine-tuning task, while significantly mitigating forgetting of general capabilities. Bold values denote the best results among these methods.

Method	GSM8K	General Capability			
		CR	IFeval	HumanEval	Avg.
Llama-2-7B-Chat	13.7	65.6	41.4	24.3	43.8
Default FT	**48.4**	62.8	30.7	15.6	36.4
HFT	46.9	63.4	31.8	20.0	38.4
LoRA	45.3	63.9	35.6	21.0	40.2
MoFO	47.1	**64.0**	**37.1**	**21.7**	**40.9**



E.4 TRANING PROCESS OF MOFO

In this subsection, we analyze the differences between the training processes of MoFO and the default SFT.

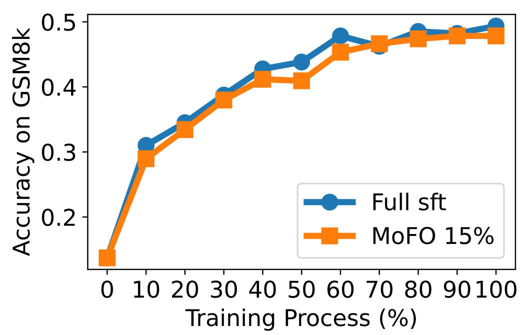


Figure 9: The GSM8K accuracy achieved during the fine-tuning of Llama-2-7B on the MetaMathQA dataset. The update fraction of MoFO is $\alpha\% = 15\%$.

Following the setting in Section 4.2, we present the GSM8K accuracy achieved during the fine-tuning of Llama-2-7B on the MetaMathQA dataset with different methods in Figure 9. The results demonstrate that the MoFO method can achieve training effectiveness comparable to the default fine-tuning approach.

E.5 COMPARISON WITH MORE FINE-TUNING METHODS

In this subsection, we compare our proposed method with the Heterogeneous Model Averaging (HMA) (Lin et al., 2024). HMA approach evenly divides the LLM into three parts—the input part, the middle part, and the output part—and averages these parts with different ratios. To facilitate a comprehensive comparison, following the setting in Section 4.2, we evaluate the fine-tuning and forgetting mitigation performance for different HMA strategies. We select 15 different combinations of averaging ratios for different parts as follows: {(0.05, 0.2, 0.35), (0.1, 0.2, 0.3), (0.2, 0.2, 0.2), (0.3, 0.2, 0.1), (0.35, 0.2, 0.05), (0.3, 0.5, 0.7), (0.4, 0.5, 0.6), (0.5, 0.5, 0.5), (0.6, 0.5, 0.4), (0.7, 0.5, 0.3), (0.65, 0.8, 0.95), (0.7, 0.8, 0.9), (0.8, 0.8, 0.8), (0.9, 0.8, 0.7), (0.95, 0.8, 0.65)}. We plot the results to construct a Pareto front in Figure 10.

Results show that our proposed method, MoFO achieves a more effective Pareto front compared to the baselines.

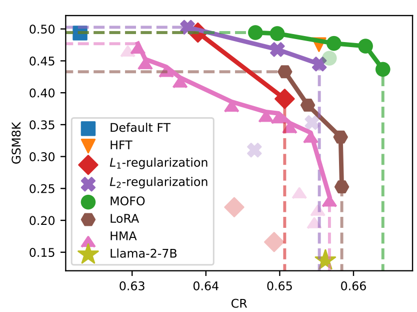


Figure 10: The performance on the math task (GSM8K) and the scores in Commonsense Reasoning of Llama-2-7B after fine-tuning on the MetaMathQA dataset. The results show that the MoFO algorithm achieves a better Pareto front. The pink triangle represents the model obtained through HMA.