LoRA+

For *Efficient* Finetuning

Presented by: Aditya Morolia

May 2, 2026

Efficient, Stable Feature Learning at Infinite Width

Goals

- Identify the conditions under which LoRA finetuning can induce ‘stable’ and ‘efficient’ feature learning.

Goals

- Identify the conditions under which LoRA finetuning can induce ‘stable’ and ‘efficient’ feature learning.
- Understand the dynamics of LoRA finetuning in the infinite-width limit.

Goals

- Identify the conditions under which LoRA finetuning can induce 'stable' and 'efficient' feature learning.
- Understand the dynamics of LoRA finetuning in the infinite-width limit.
- Lora is inefficient as $n \to \infty$.

Goals

- Identify the conditions under which LoRA finetuning can induce 'stable' and 'efficient' feature learning.
- Understand the dynamics of LoRA finetuning in the infinite-width limit.
- Lora is inefficient as $n \to \infty$.
- Propose a modified LoRA algorithm, LoRA+, that achieves stable and efficient feature learning in the infinite-width limit.

Goals

- Identify the conditions under which LoRA finetuning can induce 'stable' and 'efficient' feature learning.
- Understand the dynamics of LoRA finetuning in the infinite-width limit.
- Lora is inefficient as $n \to \infty$.
- Propose a modified LoRA algorithm, LoRA+, that achieves stable and efficient feature learning in the infinite-width limit.

Notation

- Single trainable adapter layer $A \in \mathbb{R}^{r \times n}$, $B \in \mathbb{R}^{n \times r}$.

Notation

- Single trainable adapter layer $A \in \mathbb{R}^{r \times n}, B \in \mathbb{R}^{n \times r}$.
- $A_{ij} \sim \mathcal{N}(0, \sigma_A^2),\ B_{ij} \sim \mathcal{N}(0, \sigma_B^2)$.

Notation

- Single trainable adapter layer $A \in \mathbb{R}^{r \times n}, B \in \mathbb{R}^{n \times r}$.
- $A_{ij} \sim \mathcal{N}(0, \sigma_A^2)$, $B_{ij} \sim \mathcal{N}(0, \sigma_B^2)$.
- Action: $Z_o = W^* Z_i + \frac{\alpha}{r} BAZ_i$.

Notation

- Single trainable adapter layer $A \in \mathbb{R}^{r \times n}, B \in \mathbb{R}^{n \times r}$.
- $A_{ij} \sim \mathcal{N}(0, \sigma_A^2)$, $B_{ij} \sim \mathcal{N}(0, \sigma_B^2)$.
- Action: $Z_o = W^* Z_i + \frac{\alpha}{r} BAZ_i$.
- $Z_A = AZ_i$, $Z_B = BZ_A$.

Notation

- Single trainable adapter layer $A \in \mathbb{R}^{r \times n}, B \in \mathbb{R}^{n \times r}$.
- $A_{ij} \sim \mathcal{N}(0, \sigma_A^2)$, $B_{ij} \sim \mathcal{N}(0, \sigma_B^2)$.
- Action: $Z_o = W^* Z_i + \frac{\alpha}{r} BAZ_i$.
- $Z_A = AZ_i$, $Z_B = BZ_A$.
- Either $\sigma_A^2 = \Theta(n^{-1})$, $\sigma_B^2 = 0$, or $\sigma_A^2 = 0$, $\sigma_B^2 = \Theta(1)$.

Notation

- Single trainable adapter layer $A \in \mathbb{R}^{r \times n}$, $B \in \mathbb{R}^{n \times r}$.
- $A_{ij} \sim \mathcal{N}(0, \sigma_A^2)$, $B_{ij} \sim \mathcal{N}(0, \sigma_B^2)$.
- Action: $Z_o = W^* Z_i + \frac{\alpha}{r} BAZ_i$.
- $Z_A = AZ_i$, $Z_B = BZ_A$.
- Either $\sigma_A^2 = \Theta(n^{-1})$, $\sigma_B^2 = 0$, or $\sigma_A^2 = 0$, $\sigma_B^2 = \Theta(1)$.

Remark. Last condition is for stability: $\mathrm{Var}((AZ_i)_j) = \mathcal{O}(1)$, $\mathrm{Var}((BZ_A)_j) = \mathcal{O}(1)$. Z_B is a sum of r terms, while Z_A is a sum of n terms.

Stability

Definition

We say that LoRA finetuning is stable if for all LoRA layers ℓ in the model, and all training steps t, we have

$$Z_{i,\ell}^t,\ Z_{A,\ell}^t,\ Z_{B,\ell}^t = \mathcal{O}(1) \qquad \text{as } n \to \infty.$$

Stability

Definition

We say that LoRA finetuning is stable if for all LoRA layers ℓ in the model, and all training steps t, we have

$$Z_{i,\ell}^t,\ Z_{A,\ell}^t,\ Z_{B,\ell}^t = \mathcal{O}(1) \qquad \text{as } n \to \infty.$$

- Upper bound on intermediate activations: no quantity in the computation blows up as $n \to \infty$.

Stability

Definition

We say that LoRA finetuning is stable if for all LoRA layers ℓ in the model, and all training steps t, we have

$$Z^t_{i,\ell},\ Z^t_{A,\ell},\ Z^t_{B,\ell} = \mathcal{O}(1) \qquad \text{as } n \to \infty.$$

- Upper bound on intermediate activations: no quantity in the computation blows up as $n \to \infty$.
- Implies that hyperparameters, initialization and learning rate need to be apppropriately scaled as n grows.

Stability

Definition

We say that LoRA finetuning is stable if for all LoRA layers ℓ in the model, and all training steps t, we have

$$Z_{i,\ell}^t,\ Z_{A,\ell}^t,\ Z_{B,\ell}^t = \mathcal{O}(1) \qquad \text{as } n \to \infty.$$

- Upper bound on intermediate activations: no quantity in the computation blows up as $n \to \infty$.
- Implies that hyperparameters, initialization and learning rate need to be apppropriately scaled as n grows.
- Precisely, learning rates can't be arbitarily scaled, else we get into the kernel regime (Jacot et al., 2018) where no learning occurs.

Stable Feature Learning with LoRA

Definition

We say that LoRA finetuning induces stable feature learning if it is stable, and for all LoRA layers ℓ and fine-tuning steps t, we have

$$\Delta Z_{B,\ell}^t := Z_{B,\ell}^{t+1} - Z_{B,\ell}^t = \Theta(1).$$

Stable Feature Learning with LoRA

Definition

We say that LoRA finetuning induces stable feature learning if it is stable, and for all LoRA layers ℓ and fine-tuning steps t, we have

$$\Delta Z_{B,\ell}^t := Z_{B,\ell}^{t+1} - Z_{B,\ell}^t = \Theta(1).$$

- Ensures that the network is not stuck in the lazy / kernel regime:

Stable Feature Learning with LoRA

Definition

We say that LoRA finetuning induces stable feature learning if it is stable, and for all LoRA layers ℓ and fine-tuning steps t, we have

$$\Delta Z_{B,\ell}^t := Z_{B,\ell}^{t+1} - Z_{B,\ell}^t = \Theta(1).$$

- Ensures that the network is not stuck in the lazy / kernel regime: when feature updates are $\mathcal{O}(n^{-\varepsilon})$, no learning occurs.

Stable Feature Learning with LoRA

Definition

We say that LoRA finetuning induces stable feature learning if it is stable, and for all LoRA layers ℓ and fine-tuning steps t, we have

$$\Delta Z_{B,\ell}^t := Z_{B,\ell}^{t+1} - Z_{B,\ell}^t = \Theta(1).$$

- Ensures that the network is not stuck in the lazy / kernel regime: when feature updates are $\mathcal{O}(n^{-\varepsilon})$, no learning occurs.
- Similar to Yang and Littwin, 2023. Additionally requires stability.

Anatomy of a LoRA update: 1

- Freeze all other LoRA layers. Then the input Z_i to this layer is fixed across steps.

Anatomy of a LoRA update: 1

- Freeze all other LoRA layers. Then the input Z_i to this layer is fixed across steps.

$$A_{t+1} = A_t + \Delta A_t, \qquad B_{t+1} = B_t + \Delta B_t, \qquad \Delta Z_A^t = \Delta A_t Z_i.$$

Anatomy of a LoRA update: 1

- Freeze all other LoRA layers. Then the input Z_i to this layer is fixed across steps.

$$A_{t+1} = A_t + \Delta A_t, \qquad B_{t+1} = B_t + \Delta B_t, \qquad \Delta Z_A^t = \Delta A_t Z_i.$$

$$\begin{aligned}
\Delta Z_B^t &= Z_B^{t+1} - Z_B^t \\
&= (B_t + \Delta B_t)(Z_A^t + \Delta Z_A^t) - B_t Z_A^t \\
&= \underbrace{B_t \Delta Z_A^t}_{\delta_t^1} + \underbrace{\Delta B_t Z_A^t}_{\delta_t^2} + \underbrace{\Delta B_t \Delta Z_A^t}_{\delta_t^3}.
\end{aligned}$$

Anatomy of a LoRA update: 1

- Freeze all other LoRA layers. Then the input Z_i to this layer is fixed across steps.

$$A_{t+1} = A_t + \Delta A_t, \qquad B_{t+1} = B_t + \Delta B_t, \qquad \Delta Z_A^t = \Delta A_t Z_i.$$

$$\begin{aligned}
\Delta Z_B^t &= Z_B^{t+1} - Z_B^t \\
&= (B_t + \Delta B_t)(Z_A^t + \Delta Z_A^t) - B_t Z_A^t \\
&= \underbrace{B_t \Delta Z_A^t}_{\delta_t^1} + \underbrace{\Delta B_t Z_A^t}_{\delta_t^2} + \underbrace{\Delta B_t \Delta Z_A^t}_{\delta_t^3}.
\end{aligned}$$

- δ_t^1, δ_t^2 are the linear terms; δ_t^3 is the second-order interaction terms.

Anatomy of a LoRA update: 2

Let $(B_t)_{:,j}$ be the j-th column of B_t. Since $Z_A^t = A_t Z_i \in \mathbb{R}^r$,

$$\begin{aligned} Z_B^t &= B_t Z_A^t \\ &= \sum_{j=1}^{r} (Z_A^t)_j (B_t)_{:,j} = \sum_{j=1}^{r} (A_t Z_i)_j (B_t)_{:,j}. \end{aligned}$$

Anatomy of a LoRA update: 2

Let $(B_t)_{:,j}$ be the j-th column of B_t. Since $Z_A^t = A_t Z_i \in \mathbb{R}^r$,

$$\begin{aligned} Z_B^t &= B_t Z_A^t \\ &= \sum_{j=1}^{r} (Z_A^t)_j (B_t)_{:,j} = \sum_{j=1}^{r} (A_t Z_i)_j (B_t)_{:,j}. \end{aligned}$$

Therefore,

$$\delta_t^1 = \sum_{j=1}^{r} (\Delta A_t Z_i)_j (B_t)_{:,j}, \qquad \delta_t^2 = \sum_{j=1}^{r} (A_t Z_i)_j (\Delta B_t)_{:,j}.$$

Anatomy of a LoRA update: 2

Let $(B_t)_{:,j}$ be the j-th column of B_t. Since $Z_A^t = A_t Z_i \in \mathbb{R}^r$,

$$\begin{aligned} Z_B^t &= B_t Z_A^t \\ &= \sum_{j=1}^{r} (Z_A^t)_j (B_t)_{:,j} = \sum_{j=1}^{r} (A_t Z_i)_j (B_t)_{:,j}. \end{aligned}$$

Therefore,

$$\delta_t^1 = \sum_{j=1}^{r} (\Delta A_t Z_i)_j (B_t)_{:,j}, \qquad \delta_t^2 = \sum_{j=1}^{r} (A_t Z_i)_j (\Delta B_t)_{:,j}.$$

- A changes the coefficients on the rank-r directions.
- B changes the directions themselves.

Efficient Learning

Definition

We say that LoRA fine-tuning is efficient if it is **stable**, and for all LoRA layers ℓ in the model, and all fine-tuning steps t, we have

$$\delta^1_{\ell,t},\ \delta^2_{\ell,t} = \Theta(1).$$

The main theorem!

Theorem

Assume that weight matrices A and B are trained with Adam with respective learning rates η_A and η_B

The main theorem!

Theorem

Assume that weight matrices A and B are trained with Adam with respective learning rates η_A and η_B and it holds that $g_A^t Z_i = \Theta(n)$ for the Adam-processed gradient g_A^t of A at step t.

The main theorem!

Theorem

Assume that weight matrices A and B are trained with Adam with respective learning rates η_A and η_B and it holds that $g_A^t Z_i = \Theta(n)$ for the Adam-processed gradient g_A^t of A at step t. Then, it is impossible to achieve efficiency with $\eta_A = \eta_B$.

The main theorem!

Theorem

Assume that weight matrices A and B are trained with Adam with respective learning rates η_A and η_B and it holds that $g_A^t Z_i = \Theta(n)$ for the Adam-processed gradient g_A^t of A at step t. Then, it is impossible to achieve efficiency with $\eta_A = \eta_B$. However, LoRA Finetuning is efficient with

$$\eta_A = \Theta(n^{-1}), \qquad \eta_B = \Theta(1).$$

The main theorem!

Theorem

Assume that weight matrices A and B are trained with Adam with respective learning rates η_A and η_B and it holds that $g_A^t Z_i = \Theta(n)$ for the Adam-processed gradient g_A^t of A at step t. Then, it is impossible to achieve efficiency with $\eta_A = \eta_B$. However, LoRA Finetuning is efficient with

$$\eta_A = \Theta(n^{-1}), \qquad \eta_B = \Theta(1).$$

- Assumption on the Adam-processed gradient of A makes sense in practice: one gets $g_A^t Z_i = (\text{sign}(Z_i)^\top Z_i) \times \ldots = \Theta(n)$.
- So B must move a factor n faster than A in the infinite width limit.
- A single shared learning rate cannot make both δ_t^1 and δ_t^2 to become $\Theta(1)$.

Proof Sketch

Let

$$\Delta A_t = -\eta_A g_A^t, \qquad \Delta B_t = -\eta_B g_B^t,$$

where g_A^t, g_B^t are Adam-processed gradients.

Proof Sketch

Let

$$\Delta A_t = -\eta_A g_A^t, \qquad \Delta B_t = -\eta_B g_B^t,$$

where g_A^t, g_B^t are Adam-processed gradients.
Stability gives $B_t = \Theta(1)$ and $A_t Z_i = \Theta(1)$; Adam normalization keeps $g_B^t = \Theta(1)$, and Assumption 1 gives $g_A^t Z_i = \Theta(n)$. Hence

$$\delta_t^1 = B_t \Delta A_t Z_i = \eta_A \Theta(n), \qquad \delta_t^2 = \Delta B_t A_t Z_i = \eta_B \Theta(1).$$

Proof Sketch

Let

$$\Delta A_t = -\eta_A g_A^t, \qquad \Delta B_t = -\eta_B g_B^t,$$

where g_A^t, g_B^t are Adam-processed gradients.
Stability gives $B_t = \Theta(1)$ and $A_t Z_i = \Theta(1)$; Adam normalization keeps $g_B^t = \Theta(1)$, and Assumption 1 gives $g_A^t Z_i = \Theta(n)$. Hence

$$\delta_t^1 = B_t \Delta A_t Z_i = \eta_A \Theta(n), \qquad \delta_t^2 = \Delta B_t A_t Z_i = \eta_B \Theta(1).$$

Efficient learning requires $\delta_t^1, \delta_t^2 = \Theta(1)$, so necessarily

$$\eta_A = \Theta(n^{-1}), \qquad \eta_B = \Theta(1).$$

Proof Sketch

Let

$$\Delta A_t = -\eta_A g_A^t, \qquad \Delta B_t = -\eta_B g_B^t,$$

where g_A^t, g_B^t are Adam-processed gradients.
Stability gives $B_t = \Theta(1)$ and $A_t Z_i = \Theta(1)$; Adam normalization keeps $g_B^t = \Theta(1)$, and Assumption 1 gives $g_A^t Z_i = \Theta(n)$. Hence

$$\delta_t^1 = B_t \Delta A_t Z_i = \eta_A \Theta(n), \qquad \delta_t^2 = \Delta B_t A_t Z_i = \eta_B \Theta(1).$$

Efficient learning requires $\delta_t^1, \delta_t^2 = \Theta(1)$, so necessarily

$$\eta_A = \Theta(n^{-1}), \qquad \eta_B = \Theta(1).$$

Then

$$\delta_t^3 = \Delta B_t \Delta A_t Z_i = \eta_A \eta_B \Theta(n) = \Theta(1).$$

- If $\eta_A = \eta_B = \eta$, we would need $\eta = \Theta(n^{-1})$ and $\eta = \Theta(1)$ simultaneously: contradiction.

Thank you!