**- LoRA-FA**

Theoretically, inspired by [1], freezing $A$ yields regression on projected features, while freezing $B$ yields regression on projected outputs. Specifically, when fixing $A = A_0$:

$$B^* = W^* X A_0^\top \, (A_0 X A_0^\top)^{-1}$$

When fixing $B = B_0$:

$$A^* = B_0^\top W^*$$

We simplify the representation of the input data distribution as $X$, and let $A^*$, $B^*$, $W^*$ be the optimal objectives. It is not difficult to find that, compared to $B^*$, $A^*$ does not consider the feature distribution in the input data. This is the fundamental reason for the asymmetry between $A$ and $B$.

From LoRA's calculation method, we can find that LoRA and LoRA-FA actually form a system of linear equations. In fact, the optimal $A^*$ and $B^*$ in the original LoRA can be transformed as:

$$B^* A^* X = B^* A^* X_l X_r = (B^* A^* X_l (A_0 X_l)^{-1}) A_0 X$$

Here, we suppose $X$ (at full rank; for each $X$, there exists an $X_r = X_l^{-1} X \in \mathbb{R}^r$) can be decomposed into $X_l X_r$. Intuitively, this suggests that for any $A_0$, we can always have:

$$B = B^* A^* X_l (A_0 X_l)^{-1}$$

This is the basis for why LoRA-FA can be effective.

**- LoRA-FA vs. LoRA-FB (Fix-$B$)**

We simplify each loss as below:

$$\mathcal{L}_{\text{LoRA-FA}} = d_{\text{out}} \sigma^2 + \operatorname{Tr}[W^* X {W^*}^\top] - \operatorname{Tr}[A_0 X {W^*}^\top \, W^* X A_0^\top \, (A_0 X A_0^\top)^{-1}]$$

$$\mathcal{L}_{\text{LoRA-FB}} = d_{\text{out}} X^2 + \operatorname{Tr}[W^* X {W^*}^\top] - \operatorname{Tr}[B_0^\top \, W^* X {W^*}^\top \, B_0]$$

In this way, when $r \ll d$:

$$\mathcal{L}_{\text{LoRA-FA}} \leq \mathcal{L}_{\text{LoRA-FB}}$$

This suggests that FB serves as the upper bound for FA, which is also validated in previous experiments.

**- LoRA-FA vs. LoRA**

Why can LoRA-FA achieve comparable performance to LoRA? Let's approach this from the perspective of fitting ability. Since Transformers fine-tuning is a next-token prediction problem, it conforms to the most basic generalization error principles [2].

For any $w \in \mathcal{W}$, the empirical risk $L_S(w)$ can be written in terms of $f$ as:

$$L_S(w) = f(S, w)$$

Similarly, the population risk $L_\mu(w)$ is the expected value of $f(S, w)$ under the distribution of $S$:

$$L_\mu(w) = \mathbb{E}[f(S, w)]$$

Then, for a learning algorithm $P_{W|S}$, we can get its generalization error as:

$$\text{gen}(\mu, P_{W|S}) = \mathbb{E}[f(\overline{S}, \overline{W})] - \mathbb{E}[f(S, W)]$$

where $\overline{S}$ and $\overline{W}$ are independent copies of $S$ and $W$, and the joint distribution of $S$ and $W$ is given by $P_{S,W} = \mu^{\otimes n} \otimes P_{W|S}$.

If the loss function is $\sigma$-sub-Gaussian under $\mu$ for all $w \in \mathcal{W}$, then the expected generalization error is bounded by:

$$\left|\mathrm{gen}(\mu, P_{W|S})\right| \leq \sqrt{\frac{2\sigma^2}{n} I(S;W)}$$

where $I(S;W)$ denotes the mutual information between the training dataset $S$ and the learned parameters $W$.

From this, we can further derive the generalization error bounds for LoRA and LoRA-FA, as below:

$$|\mathrm{gen}(\mu, \mathrm{LoRA})| \leq \sqrt{\frac{2rq\sigma^2 \ln 2}{n} \sum_{i \in \mathcal{I}} \left(d_{\mathrm{in}}^{(i)} + d_{\mathrm{out}}^{(i)}\right)}$$

$$|\mathrm{gen}(\mu, \mathrm{LoRA\text{-}FA})| \leq \sqrt{\frac{2rq\sigma^2 \ln 2}{n} \sum_{i \in \mathcal{I}} d_{\mathrm{out}}^{(i)}}$$

In this way, for common settings with $r$ remaining the same for both LoRA and LoRA-FA, LoRA-FA reduces half the trainable parameters by freezing $A$ compared to LoRA; its generalization error bound is only $\sqrt{2}$ smaller than that of LoRA. This suggests that to make LoRA-FA comparable to LoRA, we can simply double $r$.

More importantly, from Figure 4 in our paper, the memory consumption of LoRA-FA is not sensitive to the rank (since it erases the activation of $A$); however, LoRA consumes significant activation memory. This ensures that LoRA-FA can still reduce memory consumption while maintaining the same performance as LoRA.

**In Summary**

We can conclude as follows:

- **LoRA-FA has comparable convergence ability to LoRA.**
- **Compared with Fix-$B$, LoRA-FA can achieve better performance.**
- **Compared with LoRA, LoRA-FA can simply increase $r$ to achieve the same performance and significantly reduce memory consumption.**

[1] Asymmetry in Low-Rank Adapters of Foundation Models

[2] Information-theoretic analysis of generalization capability of learning algorithms