北京大学
PEKING UNIVERSITY

Parameter Effecient Tuning

Raise a Child in Large Language Model: Towards Effective and Generalizable Fine-tuning

ACL,2021 cite=135

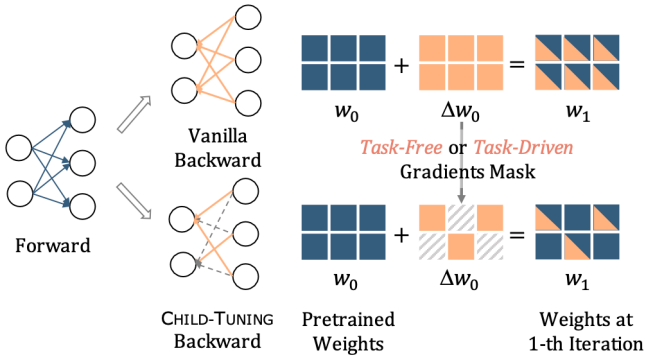


Figure 1: The illustration of CHILD-TUNING. *Left*: It forwards on the whole network while backwarding on a subset of network (i.e., child network). *Right*: To achieve this, a task-free or task-driven mask is performed on the gradients of the non-child network, resetting them to zero (grey diagonal grids).

$$\mathbb{E}[\mathbf{\Delta w}] = -\eta \frac{\partial \mathcal{L}}{\partial \mathbf{w}} \tag{6}$$

$$\Sigma[\mathbf{\Delta w}] = \frac{\eta^2 \sigma_{\mathbf{g}}^2 \mathbf{I}_k}{p_F |\mathcal{B}|} + \frac{(1-p_F)\eta^2 diag\{\frac{\partial \mathcal{L}}{\partial \mathbf{w}}\}^2}{p_F} \tag{7}$$

其中第 ((i, j)) 个元素是 (f) 对第 (i) 个变量和第 (j) 个变量的二阶偏导数，即：

$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Theorem 2. *Suppose* $\mathbf{w}_0$ *denotes the pretrained parameter;* k *is the number of parameters;* $\mathbf{w}$ *denotes the local minima the algorithm converges to;* ρ *is the greatest eigenvalue of the Hessian matrix on* $\mathbf{w}$*, which indicates the sharpness. If* $\mathbf{\Delta w} \sim N(\mathbf{0}_k, \sigma^2 \mathbf{I}_k)$*, when the following bound holds, the algorithm can converge to the local minima* $\mathbf{w}$ *with high probability,*

$$\rho \leq O(\frac{1}{\sigma^2}) \tag{8}$$

Suppose the prior over parameters after training is $P = N(\mathbf{w}_0, \sigma_0^2 \mathbf{I}_k)$*, the following generalization error bound holds with high probability,*

$$bound(\mathbf{w}) \leq O(\frac{k\sigma_0^2 - \|\mathbf{w}-\mathbf{w}_0\|^2}{\sigma^2}) + \mathcal{R} \tag{9}$$

where $\mathcal{R}$ *is a term not determined by* σ.

在寻找函数的极值时，黑塞矩阵可以帮助判断一个临界点是否为极小值或极大值。具体来说，如果黑塞矩阵在某点处是正定的，那么该点是局部极小值；如果是负定的，则是局部极大值；

黑塞矩阵的特征值表示函数在各个方向上的曲率。最大的特征值 表示在某一点上，函数曲率最大的方向

Raise a Child in Large Language Model: Towards Effective and Generalizable Fine-tuning

ACL,2021 cite=135

$$M_t^{(i)} = \begin{cases} 1, & \mathbf{w}_t^{(i)} \in \mathcal{C}_t \\ 0, & \mathbf{w}_t^{(i)} \notin \mathcal{C}_t \end{cases}$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \frac{\partial \mathcal{L}(\mathbf{w}_t)}{\partial \mathbf{w}_t} \odot \boldsymbol{M}_t$$

Fisher信息矩阵, 一种衡量参数估计量精确度的方法，用于描述参数估计的信息量

$$\mathbf{F}(\mathbf{w}) = \mathbb{E}\left[\left(\frac{\partial \log p(y|\mathbf{x};\mathbf{w})}{\partial \mathbf{w}}\right)\left(\frac{\partial \log p(y|\mathbf{x};\mathbf{w})}{\partial \mathbf{w}}\right)^{\top}\right]$$

$$p_D = \frac{|\mathcal{C}|}{|\mathcal{C}| + |\overline{\mathcal{C}}|}$$

Algorithm 1 CHILD-TUNING for Adam Optimizer

Require: $\mathbf{w}_0$: initial pretrained weights; $\mathcal{L}(\mathbf{w})$: stochastic objective function with parameters $\mathbf{w}$; η: learning rate; $\beta_1, \beta_2 \in [0, 1)$: exponential decay rates for the moment estimates;

initialize timestep $t \leftarrow 0$, first moment vector $m_0 \leftarrow 0$, second moment vector $v_0 \leftarrow 0$
while not converged **do**
  $t \leftarrow t + 1$
  // Get gradients
  $\mathbf{g}_t \leftarrow \frac{\partial \mathcal{L}(\mathbf{w}_t)}{\partial \mathbf{w}_t}$
  // Get task-free/task-driven child network
  $\mathcal{C}_t \leftarrow$ GetChildNetwork()
  // Generate a corresponding gradient mask
  $\boldsymbol{M}_t \leftarrow$ GenerateMask($\mathcal{C}_t$)
  // Employ mask for gradients
  $\mathbf{g}_t \leftarrow \mathbf{g}_t \odot \boldsymbol{M}_t$
  $\mathbf{m}_t \leftarrow \beta_1 \cdot \mathbf{m}_{t-1} + (1 - \beta_1) \cdot \mathbf{g}_t$
  $\mathbf{v}_t \leftarrow \beta_2 \cdot \mathbf{v}_{t-1} + (1 - \beta_2) \cdot \mathbf{g}_t^2$
  // Bias correction
  $\hat{\mathbf{m}}_t \leftarrow \mathbf{m}_t / (1 - \beta_1^t)$
  $\hat{\mathbf{v}}_t \leftarrow \mathbf{v}_t / (1 - \beta_2^t)$
  // Update weights
  $\mathbf{w}_t \leftarrow \mathbf{w}_{t-1} - \eta \cdot \hat{\boldsymbol{m}}_t / (\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon)$
end while
return $\mathbf{w}_t$

On the Effectiveness of Parameter-Efficient Fine-Tuning

AAAI,2023 cite = 73

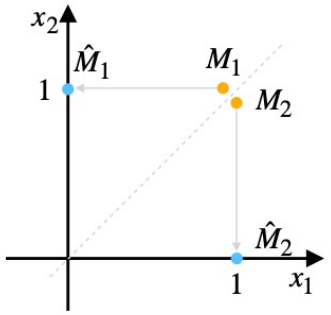


Figure 2: Projection discontinuity problem.

$$\min_{\theta} \mathcal{L}(\theta)$$
$$s.t. \ \ \|(I-M)(\theta-\theta^0)\|^2 = 0,$$

$$\bar{L} = \min_{\theta}\max_{\lambda} \mathcal{L}(\theta) + \lambda\|(I-M)(\theta-\theta^0)\|^2.$$

$$L_R = \min_{\theta} \mathcal{L}(\theta) + \|(I-M)(\theta-\theta^0)\|^2 \leq \bar{L}.$$

Pointwise Hypothesis Stability

$$\mathbb{E}_{S,i\sim U(n)}[|\ell(\mathcal{A}(S^i), z_i) - \ell(\mathcal{A}(S), z_i)|] \leq \epsilon.$$

Theorem 1 (Stability). *If the loss function ℓ is $\rho-$Lipschitz, $\mathcal{A}(S^i)$ is close to $\mathcal{A}(S)$, the Hessian matrix $\nabla^2\mathcal{L}(\mathcal{A}(S))$ at $\mathcal{A}(S)$ is positive-semidefinite with a singular value decomposition $U\,\mathrm{diag}(\Lambda)U^{-1}$, $\Lambda = \{\Lambda_1, \cdots, \Lambda_m\}$ and $\Lambda_{min} = \min\{\Lambda_1, \cdots, \Lambda_m\}$, then the expectation of the loss $\mathbb{E}_M L_R$ has a pointwise hypothesis stability as:*

$$\mathbb{E}_{S,i\sim U(n)}[|\ell(\mathcal{A}(S^i), z_i) - \ell(\mathcal{A}(S), z_i)|] \leq \frac{2\rho^2}{(\Lambda_{min} + 2(1-p))n}. \quad (6)$$

Theorem 2 (Generalization). *We denote the generalization error as $R(\mathcal{A}, S) = \mathbb{E}_z\ell(\mathcal{A}(S), z)$ and the empirical error as $\hat{R}(\mathcal{A}, S) = \frac{1}{n}\sum_{i=1}^{n}\ell(\mathcal{A}(S), z)$. Then, for some constant C, we have with probability $1-\delta$,*

$$R(\mathcal{A}, S) \leq \hat{R}(\mathcal{A}, S) + \sqrt{\frac{C^2 + \frac{24C\rho^2}{\Lambda_{min}+2(1-p)}}{2n\delta}}. \quad (7)$$

On the Effectiveness of Parameter-Efficient Fine-Tuning

AAAI,2023 cite = 73

直接获得mask的最优解，并固定以训练其他参数 θ

$$\min_{\Delta\theta, M} \mathcal{L}(\theta^0 + M\Delta\theta)$$
$$s.t. \quad \|M\|_0 = \lfloor mp \rfloor; \quad M_{ij} = 0, \forall i \neq j; \quad M_{ii} \in \{0, 1\},$$

$$\min_{\Delta\theta} \mathcal{L}(\theta^0) + \nabla \mathcal{L}(\theta^0)^{\mathrm{T}} M\Delta\theta + \frac{1}{2}(M\Delta\theta)^{\mathrm{T}} H M\Delta\theta$$
$$s.t. \quad \|M\|_0 = \lfloor mp \rfloor; \quad M_{ij} = 0, \forall i \neq j; \quad M_{ii} \in \{0, 1\}.$$

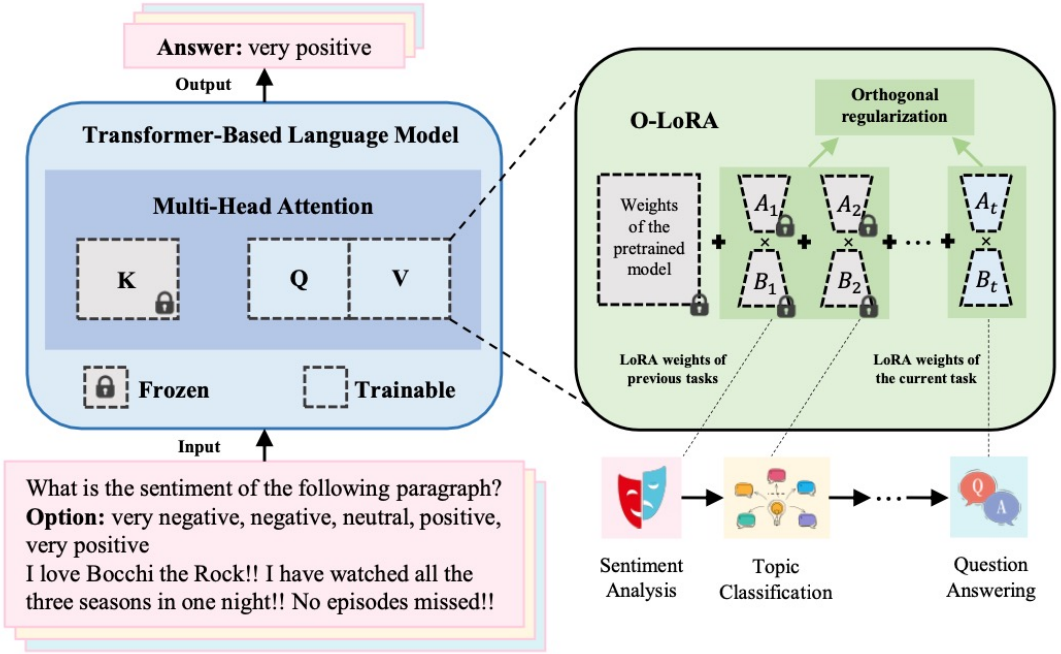
Yao et al., 2021b,a) of approximating the Hessian matrix with a diagonal matrix denoted as $H = \mathrm{diag}\{h_1, h_2, \cdots, h_n\}$. We also assume that H is positive semidefinite as the pre-trained weights

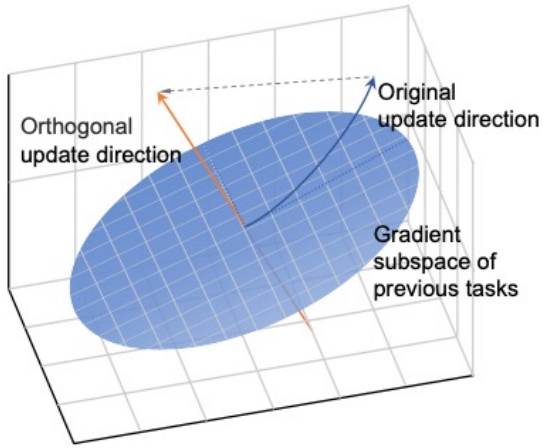
Theorem 3. *If* $\hat{M}_{ii} = \mathbb{1}(\sum_{j=1}^{m} \mathbb{1}(|\frac{\nabla \mathcal{L}(\theta^0)_i^2}{h_i}| > |\frac{\nabla \mathcal{L}(\theta^0)_j^2}{h_j}|) \geq m - \lfloor mp \rfloor)$, *where* $\nabla \mathcal{L}(\theta^0)_i$ *is the ith element of the gradient vector* $\nabla \mathcal{L}(\theta^0)$, *then*

$$\inf_{\Delta\theta} \mathcal{L}(\theta^0 + \hat{M}\Delta\theta) \leq \inf_{\substack{\Delta\theta, \|M\|_0 = \lfloor mp \rfloor; \\ M_{ij}=0, \forall i \neq j; M_{ii} \in \{0,1\}}} \mathcal{L}(\theta^0 + M\Delta\theta). \tag{9}$$

Orthogonal Subspace Learning for Language Model Continual Learning

ACL,2023 cite = 16





Orthogonal Subspace Learning for Language Model Continual Learning

ACL,2023 cite = 16

$$A_t = [a_t^1, a_t^2, ..., a_t^r]$$

$$\mathcal{U}_t = \text{span}\{a_t^1, a_t^2, ..., a_t^r\}$$

$$< u, w >= 0, \forall u \in \mathcal{U}, w \in \mathcal{W}.$$

$$O_{i,t} = A_i^T A_t = 0.$$

$$\sum_{x,y \in \mathcal{D}_t} \log p_\Theta(y \mid x) + \lambda_1 \sum_{i=1}^{t-1} L_{orth}(A_i, A_t)$$

$$L_{orth}(A_i, A_t) = \sum_{j,k} \|O_{i,t}[j,k]\|^2$$

$$W_{init} := W_{init} + \sum_{i=1}^{t} A_i B_i.$$

Controlling Text-to-Image Diffusion by Orthogonal Finetuning

NIPS,2023 cite = 40

we use hyperspherical energy to characterize the pairwise relational structure among neurons. We hypothesize that a good finetuned model should have a minimal difference in hyperspherical energy compared to the pretrained model.

$$\mathrm{HE}(W) := \sum_{i \neq j} \|\hat{w}_i - \hat{w}_j\|^{-1}$$

$$\min_{\boldsymbol{W}} \left\|\mathrm{HE}(\boldsymbol{W}) - \mathrm{HE}(\boldsymbol{W}^0)\right\| \quad \Leftrightarrow \quad \min_{\boldsymbol{W}} \left\| \sum_{i \neq j} \|\hat{\boldsymbol{w}}_i - \hat{\boldsymbol{w}}_j\|^{-1} - \sum_{i \neq j} \|\hat{\boldsymbol{w}}_i^0 - \hat{\boldsymbol{w}}_j^0\|^{-1} \right\|$$

$$\boldsymbol{z} = \boldsymbol{W}^\top \boldsymbol{x} = (\boldsymbol{R} \cdot \boldsymbol{W}^0)^\top \boldsymbol{x}, \quad \text{s.t. } \boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{R}\boldsymbol{R}^\top = \boldsymbol{I}$$

$$\boldsymbol{R} = (\boldsymbol{I} + \boldsymbol{Q})(\boldsymbol{I} - \boldsymbol{Q})^{-1}$$

$$\boldsymbol{R} = \mathrm{diag}(\boldsymbol{R}_1, \boldsymbol{R}_2, \cdots, \boldsymbol{R}_r) = \begin{bmatrix} \boldsymbol{R}_1 \in \mathrm{O}(\frac{d}{r}) & & \\ & \ddots & \\ & & \boldsymbol{R}_r \in \mathrm{O}(\frac{d}{r}) \end{bmatrix} \in \mathrm{O}(d)$$

$$\boldsymbol{z} = \boldsymbol{W}^\top \boldsymbol{x} = (\boldsymbol{R} \cdot \boldsymbol{W}^0)^\top \boldsymbol{x}, \quad \text{s.t. } \boldsymbol{R}^\top \boldsymbol{R} = \boldsymbol{R}\boldsymbol{R}^\top = \boldsymbol{I}, \ \|\boldsymbol{R} - \boldsymbol{I}\| \leq \epsilon$$

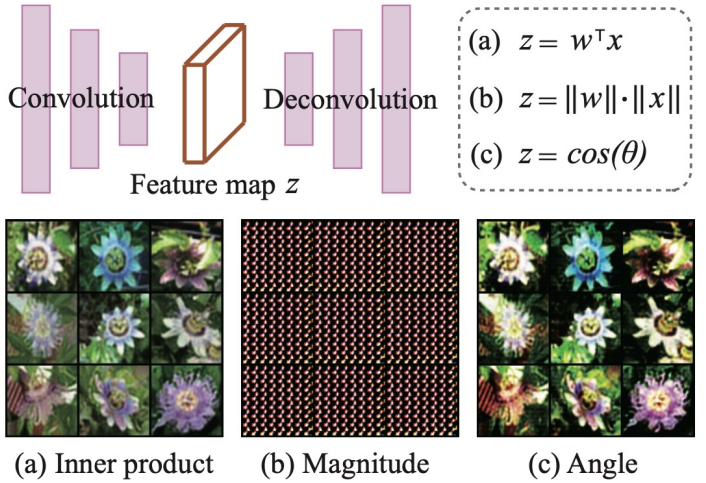
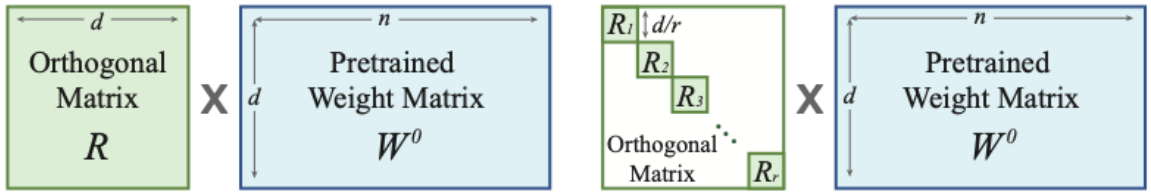


Figure 2: A toy experiment to demonstrate the importance of angular information. The autoencoder is trained in a standard way using inner product activation, and (a) shows the standard reconstruction. In testing, the angular information of neurons alone can well recover the input image, even if the autoencoder is not trained with angles.

Controlling Text-to-Image Diffusion by Orthogonal Finetuning

NIPS,2023 cite = 40



(a) Original OFT (b) OFT with block-diagonal R

Figure 4: (a) Original OFT without a diagonal structure. (b) OFT with r diagonal blocks of the same size. When $r = 1$, the case of (b) recovers the case of (a).

1. **初始化**：将正交矩阵 R 初始化为单位矩阵 I，将斜对称矩阵 Q 初始化为全零矩阵。
2. **计算输出**：使用当前的正交矩阵 R 和预训练权重矩阵 W^0 计算模型的输出。
3. **计算损失**：使用均方误差等损失函数计算预测输出与真实输出之间的损失。
4. **计算梯度并更新 Q**：计算损失函数对 Q 的梯度，并使用梯度下降法更新 Q。
5. **施加约束**：将 Q 的每个元素限制在 $[-\epsilon/2, \epsilon/2]$ 范围内，确保正交变换不会太剧烈。
6. **Cayley 变换**：使用 Cayley 变换公式将 Q 转换为正交矩阵 R，确保 R 保持正交性。
7. **微调权重**：使用更新后的正交矩阵 R 微调预训练权重矩阵 W^0，得到微调后的权重矩阵 $W_{\text{finetuned}}$。

Continual Learning with Low Rank Adaptation

arXiv preprint ,2023 cite = 2

Continual Learning with Low Rank Adaptation

Martin Wistuba
Amazon Web Services

Prabhu Teja S
Amazon Web Services

Lukas Balles
Amazon Web Services

Giovanni Zappella
Amazon Web Services

Inference As the dataset identifier D is not available at inference time, we use a simple unsupervised method [31] to infer it. We estimate k dataset prototype vectors for each dataset D at training time as follows. First, we embed each training instance using h (without LoRA modules), and run k-means on those feature embeddings. We store the k cluster centers which serve as representatives for dataset D. At inference time for an instance $\mathbf{x}$, we estimate the cluster center which is nearest to $h(\mathbf{x})$. Then, we use $f_{\hat{D}}$ to make the prediction for $\mathbf{x}$, where $\hat{D}$ is the dataset corresponding to the nearest cluster center.

Parameter Efficient Quasi-Orthogonal Fine-Tuning via Givens Rotation

preprint arXiv:2404, cite = 2

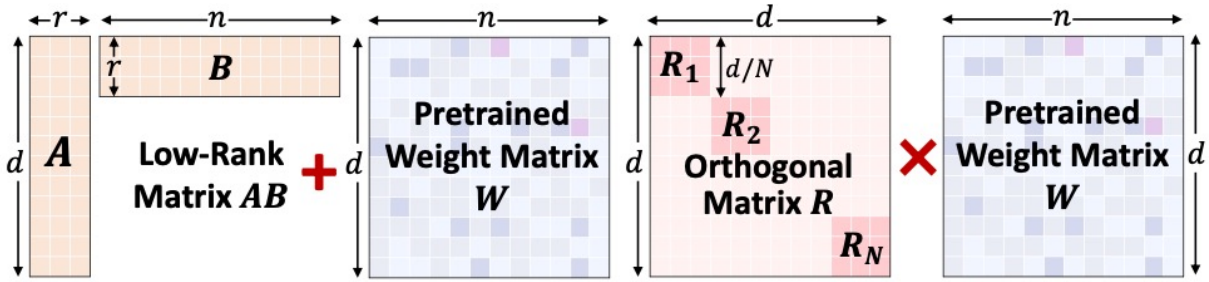


Figure 1. LoRA and OFT Reparameterization Tuning Methods.

Theorem 4.1. *Given any vector* $\boldsymbol{x} \in \mathbb{R}^d$*, there always exist* $d-1$ *Givens rotations* $\{\boldsymbol{G}(i_k, j_k; \theta_k)\}_{k=1}^{d-1}$ *that can transform* $\boldsymbol{x}$ *to any vector* $\boldsymbol{y} \in \mathbb{R}^d$ *on the same sphere with* $\boldsymbol{x}$*, i.e.,* $\prod_{k=1}^{d-1} \boldsymbol{G}(i_k, j_k; \theta_k)\boldsymbol{x} = \boldsymbol{y}$*, satisfying* $||\boldsymbol{y}||_2^2 = ||\boldsymbol{x}||_2^2$.

参数效率低 O(d^2), 论文提出一种 O(d)的方法

3.2. Givens Rotation

In numerical linear algebra, a Givens rotation (Press, 2007, Section 11.3.1) is a rotation in the plane spanned by two coordinate axes. Algebraically, a Givens rotation is represented by an orthogonal matrix of the form:

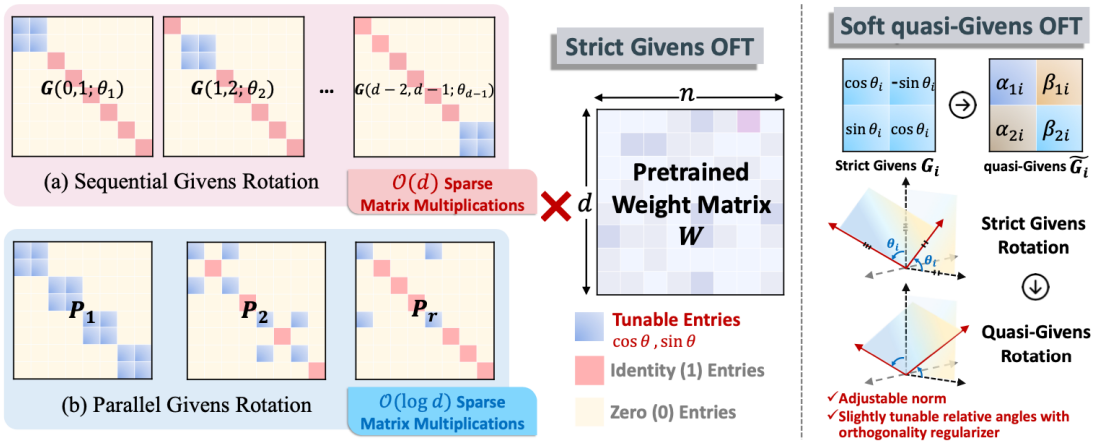
$$\boldsymbol{G}(i,j;\theta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & \cos\theta & \cdots & -\sin\theta & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & \sin\theta & \cdots & \cos\theta & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}, \tag{1}$$

where $\cos\theta$ and $\sin\theta$ appear at the intersection positions of i-th and j-th rows and columns, and the other non-zero entries are at the diagonal with all 1s. In other words, the non-zero elements g_{mn} in $\boldsymbol{G}(i,j;\theta)$ are given by:

$$\begin{cases} g_{kk} = 1, & \text{for } k \neq i, j; \\ g_{kk} = \cos\theta, & \text{for } k = i, j; \\ g_{ij} = -g_{ji} = -\sin\theta. & \end{cases} \tag{2}$$

Parameter Efficient Quasi-Orthogonal Fine-Tuning via Givens Rotation

preprint arXiv:2404, cite = 2



$$\boldsymbol{G}_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix} \rightarrow \tilde{\boldsymbol{G}}_i = \begin{bmatrix} \alpha_{1i} & \beta_{1i} \\ \alpha_{2i} & \beta_{2i} \end{bmatrix} = (\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i). \tag{5}$$

$$\begin{aligned} \boldsymbol{x}^\top \tilde{\boldsymbol{G}}_i{}^\top \tilde{\boldsymbol{G}}_i \boldsymbol{x} &= \boldsymbol{x}^\top \begin{bmatrix} \alpha_{1i}^2 + \alpha_{2i}^2 & \alpha_{1i}\beta_{1i} + \alpha_{2i}\beta_{2i} \\ \alpha_{1i}\beta_{1i} + \alpha_{2i}\beta_{2i} & \beta_{1i}^2 + \beta_{2i}^2 \end{bmatrix} \boldsymbol{x} \\ &= \boldsymbol{x}^\top \begin{bmatrix} ||\boldsymbol{\alpha}_i||_2^2 & \langle \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i \rangle \\ \langle \boldsymbol{\alpha}_i, \boldsymbol{\beta}_i \rangle & ||\boldsymbol{\beta}_i||_2^2 \end{bmatrix} \boldsymbol{x}. \end{aligned} \tag{6}$$

Figure 2. Our proposed method: quasi-Givens Orthogonal Fine-Tuning (qGOFT). The left subfigure denotes the strict GOFT which applies $d-1$ Givens rotation to left-multiply with the pretrained weight matrix, where (a) and (b) depict the sequential and parallel rotation manner, respectively. The right subfigure illustrates how qGOFT works, where each Givens rotation in GOFT is substituted with a quasi-Givens matrix for norm and angular relaxation.

$$\sum_{i=1}^{d-1} (\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i)^2$$

就使用了O(d)的参数进行正则，如果提出了并行旋转策略，减少了矩阵乘法的次数就是 O($\log(d)$)

$$\boldsymbol{P}_r = \prod_{k=0}^{(d/2^r)-1} \boldsymbol{G}(2^r k, 2^{r-1}(2k+1); \theta_k^r).$$

$$\boldsymbol{h} = (\boldsymbol{R}\boldsymbol{W})^\top \boldsymbol{x} = \left(\left(\prod_{r=1}^{\log d} \boldsymbol{P}_r \right) \boldsymbol{W} \right)^\top \boldsymbol{x}.$$

Parameter Efficient Quasi-Orthogonal Fine-Tuning via Givens Rotation

preprint arXiv:2404, cite = 2

Algorithm 1 The fine-tuning and testing procedure of a pre-trained model with (q)GOFT.

Input: Training dataset $\mathcal{D}$, Frozen Pretrained model $\mathcal{M}$, Tunable quasi-Givens Rotation Matrices $\tilde{\boldsymbol{G}}_i^n = [\boldsymbol{\alpha}_i^n, \boldsymbol{\beta}_i^n]$ denoting the i-th quasi-Givens Rotation of the n-th linear layer.
Training Stage:
Initialization: $\tilde{\boldsymbol{G}}_i^n := \boldsymbol{I}$.
for each iteration **do**
    Randomly draw a mini-batch of samples from the training set $\mathcal{D}$;
    Parallel rotating the weight matrix $\boldsymbol{W}^n$ of the fine-tuned linear layers in $\mathcal{M}$ using Eq.(3)
    Conduct forward pass using Eq.(4).
    Calculate loss function $\mathcal{L} = \mathcal{L}_{tr} + \lambda \sum_i \sum_n \langle \boldsymbol{\alpha}_i^n, \boldsymbol{\beta}_i^n \rangle^2$;
    Update quasi-Givens rotation matrices $\tilde{\boldsymbol{G}}_i^n$ with $\nabla \mathcal{L}$.
end for
Test Stage:
Merge Delta Weights: Update all the tuned linear layers $\boldsymbol{W}^n$ in $\mathcal{M}$ with $\boldsymbol{W}_*^n = (\prod_r \boldsymbol{P}_r^n)\boldsymbol{W}^n$.
for each sample $\boldsymbol{x}$ in test set **do**
    Inference using original forward function with $\boldsymbol{W}_*^n$.
end for

Gradient Projection For Parameter-Efficient Continual Learning

arXiv preprint 2405, cite = 0

基于Prompt方法的证明

Proposition 1. *Starting from the old inputs from previous tasks x_t^1 have the same outputs after learning a new task, we have:*

$$f_\theta^1(p_{t+1}^1, x_t^1) = f_\theta^1(p_t^1, x_t^1). \tag{8}$$

$$Z_t^{t+1} = \begin{bmatrix} p_{t+1} \\ x_t \end{bmatrix}$$

$$A_t^{t+1} = softmax(\frac{Q_t^{t+1} K_t^{t+1^T}}{\sqrt{(\frac{d}{h})}}).$$

$$Z_t^{t+1} \cdot Z_t^{t+1^T} = Z_t^t \cdot Z_t^{t^T}.$$

$$\begin{cases} p_{t+1} p_{t+1}^T = p_t p_t^T, \\ x_t p_{t+1}^T = x_t p_t^T, \\ p_{t+1} x_t^T = p_t x_t^T. \end{cases}$$

$$\begin{aligned} p_{t+1} p_{t+1}^T &= (p_t + \Delta p)(p_t + \Delta p)^T \\ &= p_t p_t^T + p_t \Delta p^T + \Delta p p_t^T + \Delta p \Delta p^T. \end{aligned}$$

Hypothesis 1 (zero-forgetting ideal condition). *The old inputs from previous tasks have the same outputs with parameters trained at old task and parameters after learning a new task.*

$$x_t p_{t+1}^T = x_t (p_t^T + \Delta p^T) = x_t p_t^T + x_t \Delta p^T = x_t p_t^T.$$

$$\begin{cases} x_t \Delta p^T = 0, \\ p_t \Delta p^T = 0. \end{cases}$$

SVD: $x_t = U_t \Sigma_t V_t^T$. $\Sigma_t = \begin{bmatrix} \Sigma_{t,1} & O \\ O & \Sigma_{t,0} \end{bmatrix}$ $V_t = [V_{t,1}, V_{t,0}]$

$$x_t [V_{t,1}, V_{t,0}] = U_t \begin{bmatrix} \Sigma_{t,1} & O \\ O & \Sigma_{t,0} \end{bmatrix} \qquad x_t V_{t,0} = U_t \begin{bmatrix} O \\ \Sigma_{t,0} \end{bmatrix} \approx O.$$

Let $\Delta p = \Delta p V_{t,0} V_{t,0}^T$, we can get:

$$x_t \Delta p^T = x_t (\Delta p V_{t,0} V_{t,0}^T)^T = x_t V_{t,0} V_{t,0}^T \Delta p^T = O.$$

$$p_t \Delta p = p_t (V_{t,0} V_{t,0}^T \Delta p) = 0$$

Gradient Projection For Parameter-Efficient Continual Learning

arXiv preprint 2405, cite = 0

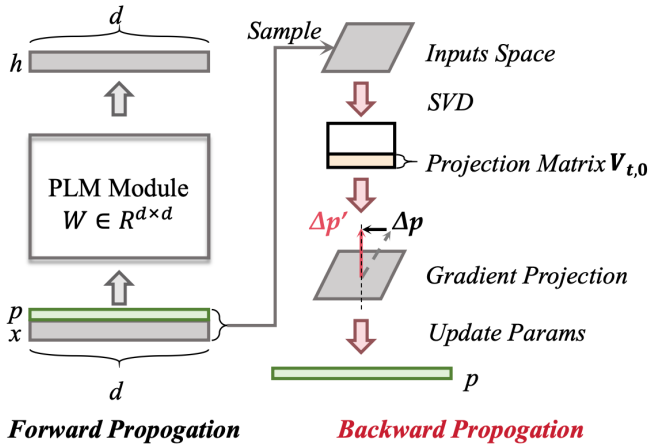


Fig. 3: Flowchart for Prompt-based continual learning

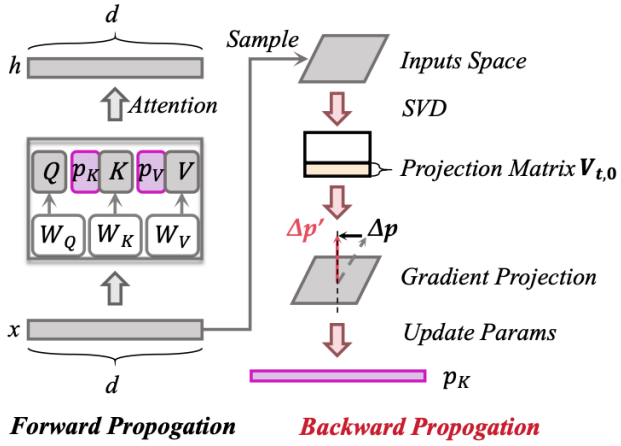


Fig. 5: Flowchart for Prefix-based continual learning.

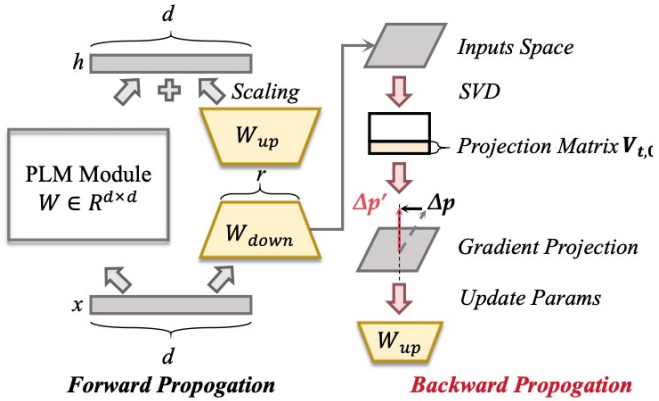


Fig. 7: Flowchart for LoRA-based continual learning.

Sample the feature space of inputs I_t

$$I_t = U_t \Sigma_t V_t^T$$

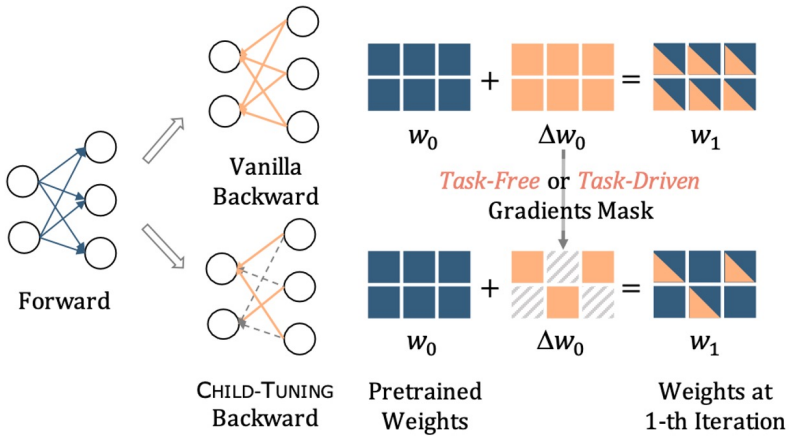
$$P_t = V_{t,0} V_{t,0}^T$$

$$\Delta E' = \Delta E V_{t,0} V_{t,0}^T$$

$$E_{t+1} = E_t + \Delta E'$$

Sparse Orthogonal Fixed Adapting (SoFA)

ACL,2021 cite=135



Vit:

1. 单任务finetune， SoFA可以和full finetune保持一样的性能
2. CL任务上：性能掉的很厉害（发现只训1个epoch，精度不掉，猜测Delta变化过大）
3. 加上限制Delta变化的L1正则化项，发现精度一点也不降
4. 但是使用prototpye预测的时候发现效果还是不行，发现使用上面的方法==只训练分类头

llm:

1. 采用T5-large（0.7B）进行实验，实验设置和O-lora一样
2. O-lora可以达到基本上不降，rouge 98%左右
3. 单任务sofa可以达到全参数ft，可以到93%左右，但是随机性很大
4. CL任务上：性能掉的很厉害

reason:

1. 模型还是太小，直接上7B?
2. 还是得进行参数选择？不能随机？（虽然参数正交，但是选的参数不敏感）
3. 还是得加正则项？
4. （第1、2篇选mask，第3、7篇正交，第4、6篇加正则化项）