Training Networks in Null Space of Feature Covariance for Continual Learning (Supplementary Material)

Anonymous CVPR 2021 submission

Paper ID 656

In this supplementary material, we first introduce additional notations, and give the proof of Lemma 1 in Appendix A. Then we discuss the reason why the parameter update satisfying Condition 2 is the descent direction in Appendix B. Finally, in Appendix C, we prove that $\langle \Delta \mathbf{w}_{t,s}, \mathbf{g}_{t,s} \rangle \geq 0$ claimed in Sec. 4.1 of the manuscript.

Notations

We first introduce additional notations here. When feeding data X_p from task $\mathcal{T}_p$ ($p \leq t$) to f with parameters $\mathbf{w}_{t,s}$, the input feature and output feature at the l-th linear layer are denoted as $X^l_{p,t,s}$ and $O^l_{p,t,s}$ respectively, then

$$O^l_{p,t,s} = X^l_{p,t,s} w^l_{t,s}, \;\; X^{l+1}_{p,t,s} = \sigma_l(O^l_{p,t,s})$$

with $X^1_{p,t,s} = X_p$. In addition, by denoting the learning rate as α, we have

$$w^l_{t,s} = w^l_{t,s-1} - \alpha \Delta w^l_{t,s-1}, l = 1, \ldots, L.$$

Appendix A

In this appendix, we show the proof of Lemma 1 in the manuscript. Lemma 1 tells us that, when we train network on task $\mathcal{T}_t$, the network retains its training loss on data X_p in the training process, if the network parameter update satisfies Eqn. (1) at each training step. We first recall Lemma 1 as follows, then give the proof.

Lemma 1. *Given the data X_p from task $\mathcal{T}_p$, and the network f with L linear layers is trained on task $\mathcal{T}_t$ ($t > p$). If network parameter update $\Delta w^l_{t,s}$ lies in the null space of $X^l_{p,t-1}$,* i.e.,

$$X^l_{p,t-1} \Delta w^l_{t,s} = 0, \tag{1}$$

at each training step s, for the l-th layer of f ($l = 1, \ldots, L$), we have $X^l_{p,t} = X^l_{p,t-1}$ and $f(X_p, \tilde{\mathbf{w}}_{t-1}) = f(X_p, \tilde{\mathbf{w}}_t)$.

Proof. The proof is based on the recursive structure of network and iterative training process. We first prove that $X^l_{p,t,1} = X^l_{p,t-1}$ and $f(X_p, \mathbf{w}_{t,1}) = f(X_p, \tilde{\mathbf{w}}_{t-1})$ hold for $s = 1$, and then illustrate that $X^l_{p,t,s} = X^l_{p,t-1}$ and $f(X_p, \mathbf{w}_{t,s}) = f(X_p, \tilde{\mathbf{w}}_{t-1})$ hold for each $s > 1$, which suggests that Lemma 1 holds.

When $s = 1$, considering that we initialize parameters $\mathbf{w}_{t,0} = \tilde{\mathbf{w}}_{t-1}$, we have

$$X^l_{p,t,0} = X^l_{p,t-1}, \;\; O^l_{p,t,0} = O^l_{p,t-1}. \tag{2}$$

Therefore, at the first layer ($l = 1$) where $X^1_{p,t,1} = X^1_{p,t,0} = X^1_{p,t-1}$ (all of them equal to X_p when $l = 1$),

$$\begin{aligned} O^1_{p,t,1} &= X^1_{p,t,1} w^1_{t,1} \\ &= X^1_{p,t,0}(w^1_{t,0} - \alpha \Delta w^1_{t,0}) \\ &= X^1_{p,t,0} w^1_{t,0} - \alpha X^1_{p,t-1} \Delta w^1_{t,0} \\ &= X^1_{p,t,0} w^1_{t,0} \\ &= O^1_{p,t,0}, \end{aligned} \tag{3}$$

where the fourth equation holds due to Eqn. (1). Furthermore, we have

$$X^2_{p,t,1} = \sigma_1(O^1_{p,t,1}) = \sigma_1(O^1_{p,t,0}) = X^2_{p,t,0} = X^2_{p,t-1}, \tag{4}$$

i.e., the input feature $X^2_{p,t,1}$ equals to $X^2_{p,t-1}$ at the second linear layer, based on which, we can recursively prove that

$$O^l_{p,t,1} = O^l_{p,t,0} = O^l_{p,t-1}$$

and

$$X^l_{p,t,1} = X^l_{p,t,0} = X^l_{p,t-1}$$

for $l = 3, \ldots, L$ by replacing $l = 1$ with $l = 2, \ldots, L$ in Eqns. (3) and (4), then we have $f(X_p, \mathbf{w}_{t,1}) = f(X_p, \tilde{\mathbf{w}}_{t-1})$.

We now have proved that $X^l_{p,t,s} = X^l_{p,t-1}$, $O^l_{p,t,s} = O^l_{p,t-1}$ ($l = 1, \ldots, L$) and $f(X_p, \mathbf{w}_{t,s}) = f(X_p, \tilde{\mathbf{w}}_{t-1})$ hold for $s = 1$. Considering the iterative training process, we can prove that

$$X^l_{p,t,s} = X^l_{p,t-1}, \;\; O^l_{p,t,s} = O^l_{p,t-1} \;\; (l = 1 \ldots, L)$$

and

$$f(X_p, \mathbf{w}_{t,s}) = f(X_p, \tilde{\mathbf{w}}_{t-1})$$

hold for $s = 2, ...,$ by repeating the above process with $s = 2, ...$

Finally, we have $X^l_{p,t} = X^l_{p,t-1}$ and $f(X_p, \tilde{\mathbf{w}}_{t-1}) = f(X_p, \tilde{\mathbf{w}}_t)$, since Lemma 1 holds for each $s \geq 1$. $\square$

Appendix B

We first recall the Condition 2 in the manuscript as follows, then prove that parameter update $\Delta\mathbf{w}_{t,s}$ satisfying condition 2 is the descent direction, *i.e.*, the training loss after updating parameters using $\Delta\mathbf{w}_{t,s}$ will decrease.

Condition 2 (plasticity)**.** *Assume that the network f is being trained on task $\mathcal{T}_t$, and $\mathbf{g}_{t,s} = \{g^1_{t,s}, \dots, g^L_{t,s}\}$ denotes the parameter update generated by a gradient-descent training algorithm for training f at training step s. $\langle \Delta\mathbf{w}_{t,s}, \mathbf{g}_{t,s}\rangle > 0$ should hold where $\langle\cdot,\cdot\rangle$ represents inner product.*

We now discuss the reason why $\Delta\mathbf{w}_{t,s}$ is the descent direction, if it satisfies condition 2. For clarity, we denote the loss for training network f as $\mathcal{L}(\mathbf{w})$ which ignores the data term with no effect. The discussion can also be found in Lemma 2 of the lecture[1].

By denoting the learning rate as α, and $h(\alpha) \triangleq \mathcal{L}(\mathbf{w}_{t,s} - \alpha\Delta\mathbf{w}_{t,s})$, according to Taylor's theorem, we have

$$h(\alpha) = h(0) + \nabla_\alpha h(0) + o(\alpha),$$

i.e.,

$$\mathcal{L}(\mathbf{w}_{t,s} - \alpha\Delta\mathbf{w}_{t,s}) = \mathcal{L}(\mathbf{w}_{t,s}) - \alpha\langle\Delta\mathbf{w}_{t,s}, \mathbf{g}_{t,s}\rangle + o(\alpha),$$

where $\frac{|o(\alpha)|}{\alpha} \to 0$ when $\alpha \to 0$. Therefore, there exists $\bar{\alpha} > 0$ such that

$$|o(\alpha)| < \alpha|\langle\Delta\mathbf{w}_{t,s}, \mathbf{g}_{t,s}\rangle|, \;\; \forall\alpha \in (0, \bar{\alpha}).$$

Together with the condition $\langle\Delta\mathbf{w}_{t,s}, \mathbf{g}_{t,s}\rangle > 0$, we can conclude that $\mathcal{L}(\mathbf{w}_{t,s} - \alpha\Delta\mathbf{w}_{t,s}) < \mathcal{L}(\mathbf{w}_{t,s})$ for all $\alpha \in (0, \bar{\alpha})$. Therefore, parameter update $\Delta\mathbf{w}_{t,s}$ satisfying condition 2 is the descent direction.

Appendix C

Here, we give the proof of $\langle\Delta\mathbf{w}_{t,s}, \mathbf{g}_{t,s}\rangle \geq 0$ with $\Delta w^l_{t,s} = U^l_2(U^l_2)^\top g^l_{t,s}$, which is claimed in Sec 4.1 of the manuscript. The proof mainly utilizes the properties of Kronecker product [1, Eqns. (2.10) and (2.13)].

$$\begin{aligned}
\langle\Delta\mathbf{w}_{t,s}, \mathbf{g}_{t,s}\rangle &= \sum_{l=1}^{L}\langle U^l_2(U^l_2)^\top g^l_{t,s}, g^l_{t,s}\rangle \\
&= \sum_{l=1}^{L}\mathrm{vec}(U^l_2(U^l_2)^\top g^l_{t,s}I)^\top\mathrm{vec}(g^l_{t,s}) \\
&= \sum_{l=1}^{L}\mathrm{vec}((U^l_2)^\top g^l_{t,s})^\top(I\otimes(U^l_2)^\top)\mathrm{vec}(g^l_{t,s}) \\
&= \sum_{l=1}^{L}\mathrm{vec}((U^l_2)^\top g^l_{t,s})^\top\mathrm{vec}((U^l_2)^\top g^l_{t,s}) \\
&\geq 0,
\end{aligned} \tag{5}$$

where $\mathrm{vec}(\cdot)$ is the vectorization of $\cdot$, I is the identity matrix and $\otimes$ is the Kronecker product.

Appendix D

We now discuss the difference between our algorithm and OWM [2] in details as follows. (1) We provide novel theoretical conditions for the stability and plasticity of network based on feature covariance. (2) The null space of ours is defined as the null space of feature covariance matrix which is easy to be accumulated after each task (refer to Q1 & Alg. 2). While the projection matrix in OWM is $\mathbf{P}_l = \mathbf{I}_l - \mathbf{A}_l(\mathbf{A}_l^\top\mathbf{A}_l + \beta_l\mathbf{I}_l)^{-1}\mathbf{A}_l^\top$ where $\mathbf{A}_l$ consists of all previous features of layer l. (3) With the coming of new tasks, our covariance matrix is incrementally updated without approximation error, while $\mathbf{P}_l$ of OWM is updated by recursive least square, where the approximation error of matrix inversion (because of the additionally introduced $\beta_l\mathbf{I}$) will be accumulated. (4) Our approach relies on a hyperparameter a in line 14 of Alg. 2, for approximating the null space of covariance, which can balance the stability and plasticity as discussed in lines 572-579 and Fig. 5. It is easy to set the hyperparameter (line 614 and Figs. 4, 5). But we find that it is hard to tune the hyperparameter β_l in OWM for each layer to balance the approximation error and computational stability. (5) Experimental comparison with OWM on three benchmarks are shown in Tabs. 1-3. The ACC of ours are 4.88%, 7.48% and 8.3% higher than OWM with comparable BWT. Please refer to Q4 for comparison on ImageNet with deeper networks. We will clarify these differences by extending the discussions in Sect. 2.

References

[1] Alexander Graham. *Kronecker products and matrix calculus with applications*. Courier Dover Publications, 2018. 2

[2] Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8):364–372, 2019. 2

[1] http://www.princeton.edu/~aaa/Public/Teaching/ORF363_COS323/F14/ORF363_COS323_F14_Lec8.pdf