MetaFGNet: Supplementary Material

Yabin Zhang, Hui Tang and Kui Jia
School of Electronic and Information Engineering
South China University of Technology, Guangzhou, China
{zhang.yabin,eehuitang}@mail.scut.edu.cn, kuijia@scut.edu.cn

This is the supplementary material of [4] to demonstrate how to evaluate gradients of the meta-learning loss with respect to model parameters.
The gradients of meta-learning loss are uniformly formulated as:

$$[\triangle(\theta_b;\mathcal{T}_j),\triangle(\theta_c^t;\mathcal{T}_j)] = \nabla_{\boldsymbol{\theta}^t}\frac{1}{|\mathcal{T}_j|}L\left(\mathcal{T}_j;\boldsymbol{\theta}^{t'}\right) = \nabla_{\boldsymbol{\theta}^{t'}}\frac{1}{|\mathcal{T}_j|}L\left(\mathcal{T}_j;\boldsymbol{\theta}^{t'}\right)\left[\boldsymbol{I}-\eta\frac{1}{|\mathcal{T}_i|}\left(\frac{\partial^2 L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^2}\right)\right], \tag{1}$$

which is easily implemented by TensorFlow [1] but a little bit more complicated with PyTorch [3]. We present to implement it using PyTorch in this supplementary material. Firstly, we expand Equ. 1 as:

$$\nabla_{\boldsymbol{\theta}^{t'}}\tfrac{1}{|\mathcal{T}_j|}L\left(\mathcal{T}_j;\boldsymbol{\theta}^{t'}\right) - \eta\nabla_{\boldsymbol{\theta}^{t'}}\tfrac{1}{|\mathcal{T}_j|}L\left(\mathcal{T}_j;\boldsymbol{\theta}^{t'}\right)\tfrac{1}{|\mathcal{T}_i|}\left(\tfrac{\partial^2 L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^2}\right),$$

where $\nabla_{\boldsymbol{\theta}^{t'}}\frac{1}{|\mathcal{T}_j|}L\left(\mathcal{T}_j;\boldsymbol{\theta}^{t'}\right)$ is directly computed via back propagation. Hence, we focus on calculating:

$$\nabla_{\boldsymbol{\theta}^{t'}}L\left(\mathcal{T}_j;\boldsymbol{\theta}^{t'}\right)\left(\frac{\partial^2 L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^2}\right). \tag{2}$$

The Hessian matrix in Equ. 2 can be transformed into:

$$\frac{\partial^2 L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^2} = \frac{\partial^2 L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial\boldsymbol{\theta}^t\partial(\boldsymbol{\theta}^t)^T} = \frac{\partial}{\partial\boldsymbol{\theta}^t}\left[\frac{\partial L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^T}\right] = \frac{\partial\boldsymbol{g}^t}{\partial\boldsymbol{\theta}^t}, \tag{3}$$

where $\theta^t \in \mathbb{R}^{1\times N}$ [1], $\boldsymbol{g}^t = \frac{\partial L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^T} \in \mathbb{R}^{N\times 1}$, $\frac{\partial^2 L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^2} \in \mathbb{R}^{N\times N}$. The ith row of $\frac{\partial^2 L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^2}$ is rewritten as $\frac{\partial\boldsymbol{g}_i^t}{\partial\boldsymbol{\theta}^t}$, which can be obtained on the basis of the chain rule.

To date, we are able to compute gradients of the meta-learning loss. However, it is computationally expensive to calculate the Hessian matrix. Thus, we propose to conduct gradient computation in a faster way.

For the sake of brevity, we abbreviate $\nabla_{\boldsymbol{\theta}^{t'}}L\left(\mathcal{T}_j;\boldsymbol{\theta}^{t'}\right)$ and $\frac{\partial^2 L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^2}$ as $\boldsymbol{G}$ and $\boldsymbol{H}$ respectively. The Equ. 2 can be rewritten as:

$$\nabla_{\boldsymbol{\theta}^{t'}}L\left(\mathcal{T}_j;\boldsymbol{\theta}^{t'}\right)\left(\frac{\partial^2 L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^2}\right) = \boldsymbol{G}\boldsymbol{H} = \sum_{i=1}^{N}\boldsymbol{G}_{1i}\cdot\boldsymbol{H}_{i\cdot} = \sum_{i=1}^{N}\boldsymbol{G}_{1i}\cdot\frac{\partial\boldsymbol{g}_i^t}{\partial\boldsymbol{\theta}^t}, \tag{4}$$

where $\boldsymbol{G}_{1i}$ indicates the ith element in the vector $\boldsymbol{G}$ and $\boldsymbol{H}_{i\cdot}$ indicates the ith row of $\boldsymbol{H}$. $\boldsymbol{G}_{1i}$ is considered as a scalar, which is not related to θ^t and thus implies $\frac{\partial\boldsymbol{G}_{1i}}{\partial\boldsymbol{\theta}^t} = 0$. Then Equ. 4 can be rewritten as:

$$\sum_{i=1}^{N}\boldsymbol{G}_{1i}\cdot\frac{\partial\boldsymbol{g}_i^t}{\partial\boldsymbol{\theta}^t} = \sum_{i=1}^{N}\frac{\partial\boldsymbol{G}_{1i}\cdot\boldsymbol{g}_i^t}{\partial\boldsymbol{\theta}^t} = \frac{\partial(\sum_{i=1}^{N}\boldsymbol{G}_{1i}\cdot\boldsymbol{g}_i^t)}{\partial\boldsymbol{\theta}^t}. \tag{5}$$

It is much faster to evaluate gradients of meta-learning loss through Equ. 5.

There also exists a simpler and faster first-order approximation [2] of Equ. 1 that drops the Hessian (i.e., assuming zero curvature around θ^t), resulting in: $\nabla_{\boldsymbol{\theta}^{t'}}\frac{1}{|\mathcal{T}_j|}L\left(\mathcal{T}_j;\boldsymbol{\theta}^{t'}\right)\left[\boldsymbol{I}-\eta\frac{1}{|\mathcal{T}_i|}\left(\frac{\partial^2 L(\mathcal{T}_i;\boldsymbol{\theta}^t)}{\partial(\boldsymbol{\theta}^t)^2}\right)\right] \approx \nabla_{\boldsymbol{\theta}^{t'}}\frac{1}{|\mathcal{T}_j|}L\left(\mathcal{T}_j;\boldsymbol{\theta}^{t'}\right)$.

[1] Model parameters in matrix form can be viewed as vectors.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283, 2016. 1

[2] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017. 1

[3] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017. 1

[4] Y. Zhang, H. Tang, and K. Jia. Fine-grained visual categorization using meta-learning optimization with sample selection of auxiliary data. *arXiv preprint arXiv:1807.10916*, 2018. 1