Supplementary File for Large-Scale Meta-Learning with Continual Trajectory Shifting

JaeWoong Shin [*1] **Hae Beom Lee** [*1] **Boqing Gong** [2 1] **Sung Ju Hwang** [1 3]

This supplementary file consists of the following contents:

- **Section A:** We show that the type of inner-optimizer (e.g. SGD with momentum or Adam) can largely affect the quality of the initialization parameters at convergence.
- **Section B:** We visualize the effect of trajectory shifting with the synthetic experiments.
- **Section C:** We provide detailed description of the experimental setup for each experiment in the main paper, including the synthetic experiment, the image classification, the ImageNet experiment, and the empirical error analysis.
- **Section D:** We derive Eq. (5) in the main paper, which is the complexity of the approximation error caused by the proposed continual trajectory shifting.
- **Section E:** We prove that we can use the same shifting rule even with the momentum optimizer and weight decaying.

A. Effect of Inner-optimizer Type

We briefly discuss if we can add in weight decay or change the type of optimizers in defining $U_k(\phi)$, without changing the results of Eq. (5) in the main paper. We also discuss which optimizer works relatively better over the others. The type of inner-optimizer is highly relevant to the quality of ϕ at convergence. Specifically, inner-optimizers with faster convergence result in faster meta-convergence as well, showing the strong dependency between the inner- and meta- optimization. Also, inner-optimizers that exhibit oscillating behavior helps the meta learner to escape from bad local minima. Momentum optimizer (Sutskever et al., 2013) is the one with all those properties.

Momentum and weight decay Given the momentum $\mu \in [0, 1]$ and weight decay $\lambda \geq 0$, we can show that we can apply the same shifting rule $\theta_k \leftarrow \theta_k + \Delta_k$ introduced in the main paper. This will only result in higher approximation error compared to the vanilla SGD case. See Section E for the derivation of the following results:

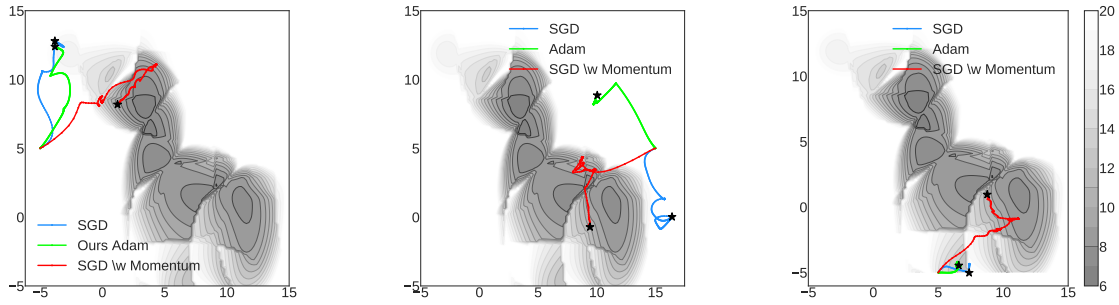
$$U_k\left(\phi + \sum_{i=1}^{k-1} \Delta_i\right) = U_1(\cdots U_1(U_1(\phi) + \Delta_1) \cdots + \Delta_{k-1}) + O\left(\beta\alpha(h + 2\lambda)k^2 + \beta^2 k\right) \tag{1}$$

for $k \geq 2$.

Adam Unfortunately, the analogous derivation for Adam optimizer (Kingma & Ba, 2014) requires to differentiate very complicated expression involving element-wise square root division. Therefore, although we may use the same shifting rule $\theta_k \leftarrow \theta_k + \Delta_k$ together with the Adam optimizer, we cannot expect that the approximation error will be bounded in any reasonable way. However, we do not have to consider Adam as an inner-optimizer in context of meta-training because oscillating property of momentum optimizer is preferable over stable learning trajectory provided by Adam optimizer. In our synthetic experiment, we tried applying Adam, but obtained much worse initial model parameters than using momentum. See Figure 1 for the actual meta-learning trajectories obtained with the various optimizers.

[*]Equal contribution [1]Graduate School of AI, KAIST, South Korea [2]Google, LA [3]AITRICS, South Korea. Correspondence to: Sung Ju Hwang <sjhwang82@kaist.ac.kr>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

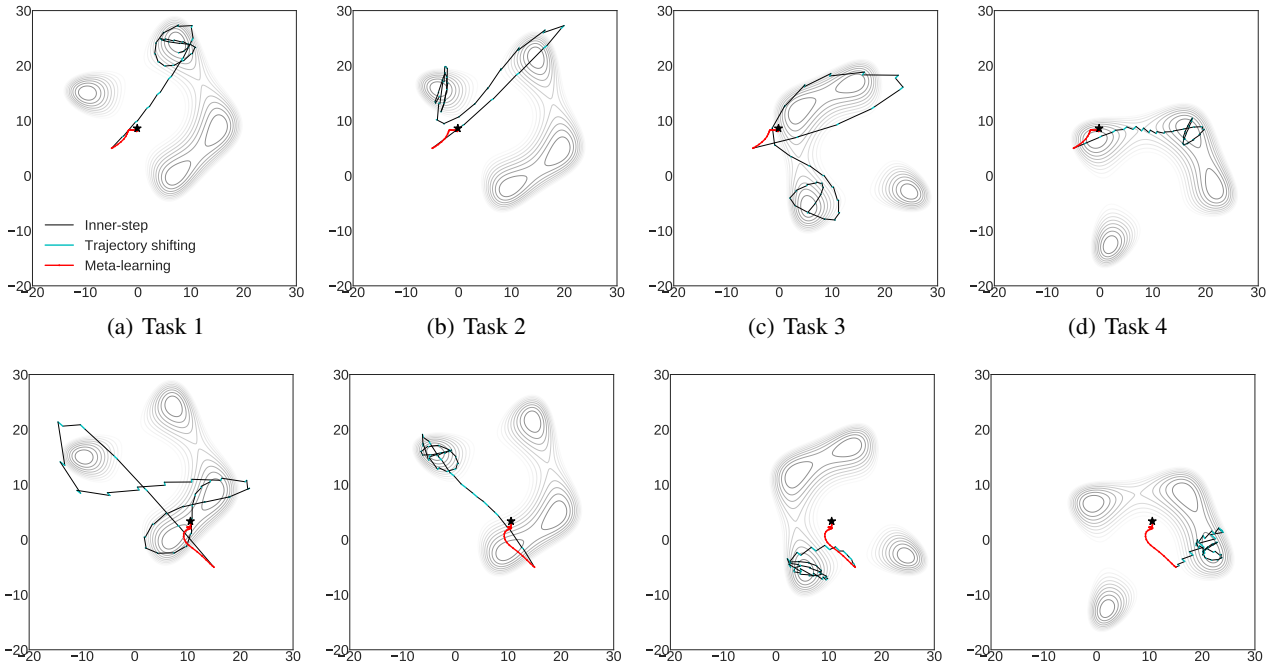


(a) Starting point: (-5, 5) (b) Starting point: (15, 5) (c) Starting point: (5, -5)

Figure 1. **Lines:** Meta-training trajectory of our method with various types of inner-optimizers. **Background contour:** Task-average loss after 100 gradient steps. The darker the background contour, the better quality of the initialization point.

B. Visualization of Trajectory Shifting

In Figure 2, we visualize the actual trajectory shifting with the synthetic experiments. We can see how each of the inner-learning trajectories is interleaved with a sequence of meta-updates.



(a) Task 1 (b) Task 2 (c) Task 3 (d) Task 4

(e) Task 1 (f) Task 2 (g) Task 3 (h) Task 4

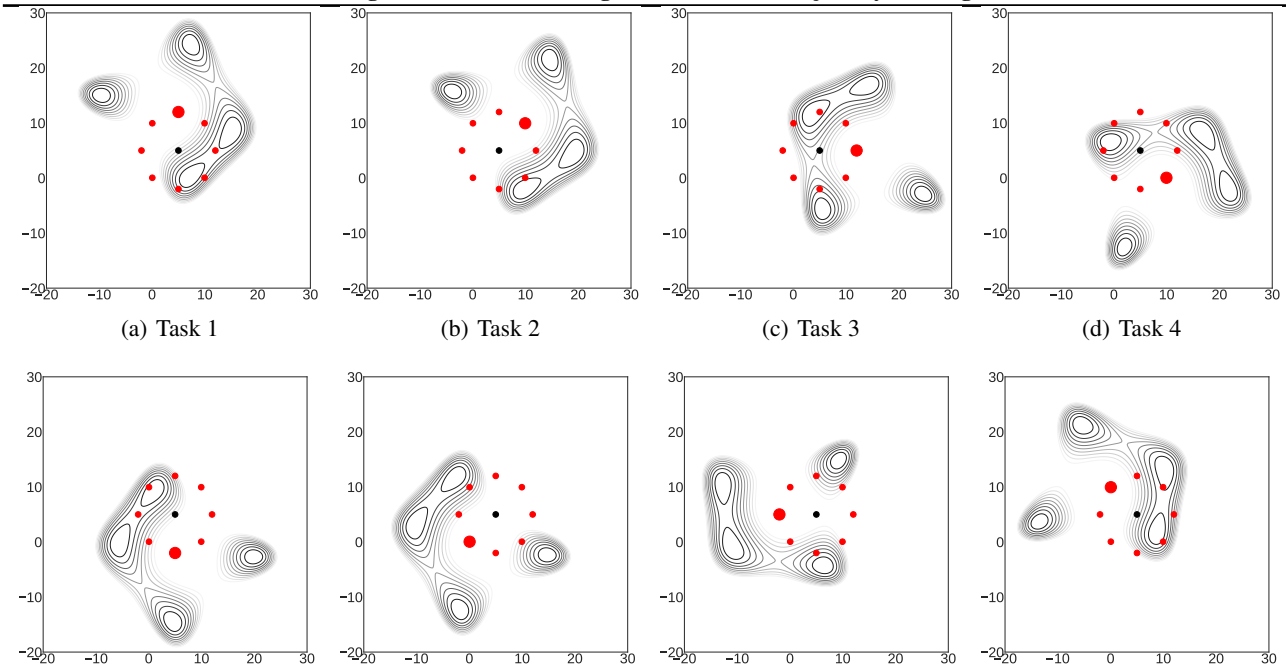
Figure 2. Visualization of the trajectory shifting with the four tasks (Task 1 - Task 4) from the synthetic experiments. **Top row:** starting point: (-5, 5). **Bottom row:** starting point: (15, 5).

C. Experimental Setup

In this section, we provide the detailed experimental setup for the synthetic experiments, the image classifications, the ImageNet experiments, and the empirical error analysis.

C.1. Synthetic experiments

We visualize in Figure 3 the loss surfaces of all the eight tasks used for the synthetic experiments.



(a) Task 1 (b) Task 2 (c) Task 3 (d) Task 4

(e) Task 5 (f) Task 6 (g) Task 7 (h) Task 8

Figure 3. Loss surfaces of the eight tasks.

C.2. Image classifications

We provide some additional information about the experimental setup for the image classification experiments.

Table 1. **Meta-training datasets** for the image classification experiments.

Dataset	# training instances	# test instances	# classes	Image size	Note
Tiny ImageNet split 1 (tin)	50,000	5,000	100	64	Class 1-100
Tiny ImageNet split 2 (tin)	50,000	5,000	100	64	Class 101-200
CIFAR100 (Krizhevsky et al., 2009)	50,000	10,000	100	32	
Stanford Dogs (Khosla et al., 2011)	11,999	8,580	120	84	
Aircraft (Maji et al., 2013)	6,667	3,333	100	84	
CUB (Wah et al., 2011)	5,994	5,794	200	84	
Fashion-MNIST (Xiao et al., 2017)	60,000	10,000	10	28	Grey-scale
SVHN (Netzer et al., 2011)	73,257	26,032	10	32	

Table 2. **Target datasets** for the image classification experiments.

Dataset	# training instances	# test instances	# classes	Image size	Note
Stanford Cars (Krause et al., 2013)	8,144	8,041	196	84	
QuickDraw (Ha & Eck, 2017)	34,500	34,500	345	28	Grey-scale
VGG Flowers (Nilsback & Zisserman, 2008)	2,040	6,149	102	84	
VGG Pets (Parkhi et al., 2012)	3,680	3,669	37	84	
STL10 (Coates et al., 2011)	5,000	8,000	10	32	

Table 3. The value of β used for the image classification experiments.

	K		
Method	10	100	1000
FOMAML (Finn et al., 2017)	0.5	0.2	0.1
Leap (Flennerhag et al., 2019)	0.1	0.1	0.1
Reptile (Nichol et al., 2018)	5	2	1
Ours	1	0.1	0.01

- See Table 1 for more information about the datasets used for meta-training and Table 2 for meta-testing.

- We carefully tuned the meta-learning rate β for all the meta-learning baselines. Notably, we found that the optimal β should increase as we reduce the length of inner-optimization trajectory K. See Table 3 for the actual value of β we used in the experiments.

- The last fully connected layer (classifier) is a part of θ, but not included in ϕ. Batch norm parameters (i.e. scale and shift) are included in both θ and ϕ. However, batch statistics (i.e. running mean and running variance) are neither a part of θ nor ϕ.

- Recall from the Algorithm 1 and 2 in the main paper that we repeat the inner-optimization process M times, and we reset the task-specific parameters before starting each process. Note that we **do not** reset the following information: the statistics for the optimizer, the parameters for the last fully connected layer, and the batch norm statistics. Conceptually, it would be natural to reset the above information as well because we do not transfer them to meta-testing. However, we found that it makes no difference in terms of meta-testing performance.

C.3. ImageNet experiments

We next provide the detailed description about the datasets used for the ImageNet experiments in Figure 4, Table 4, and Table 5.

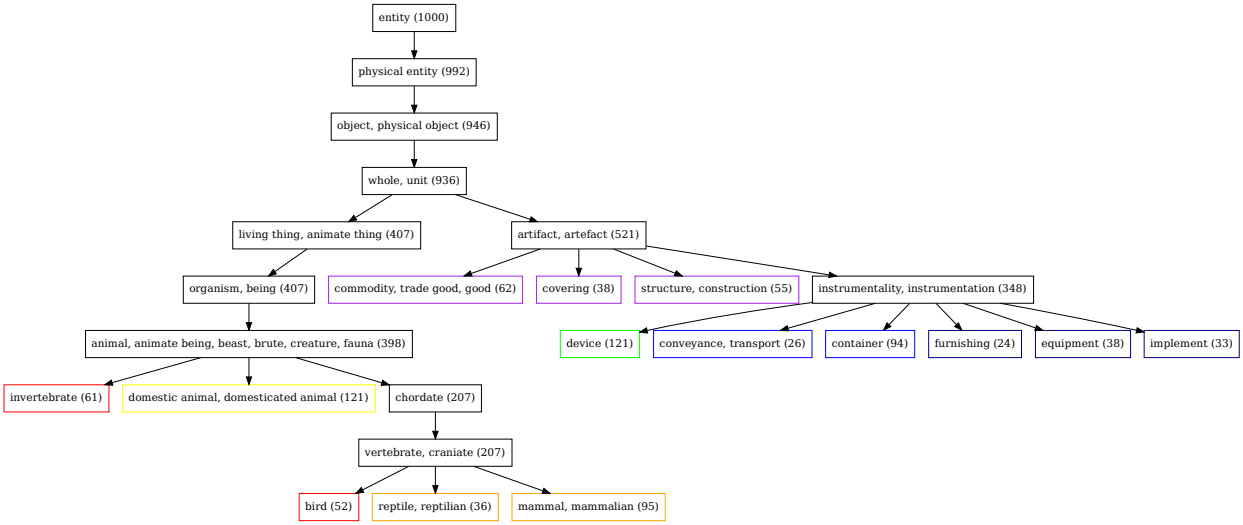


Figure 4. **ImageNet splits based on the WordNet hierarchy**. Each node correspones to a WordNet label and the number of its subclasses is shown in the parenthesis. The nodes of the same split are shown in the same color. We do not show the nodes of the last split 8 for better visualization.

Table 4. ImageNet splits for **meta-training**

Split	# training instances	# test instances	# classes	WordNet label (# classes)
1	146,741	5,650	113	bird (52) invertebrate (61)
2	169,582	6,550	131	reptile (36) mammal (95)
3	151,773	6,050	121	domestic animal(121)
4	154,047	6,050	121	device (121)
5	154,373	6,000	120	conveyance, transport (26) container (94)
6	121,344	4,750	95	furnishing (24) equipment (38) implement (33)
7	198,314	7,750	155	commodity, trade good, good (62) covering (38) structure, construction (55)
8	184,993	7,200	144	etc

Table 5. **Target datasets** for the ImageNet experiments.

Dataset	# training instances	# test instances	# classes	Image size	Note
CIFAR100 (Krizhevsky et al., 2009)	50,000	10,000	100	128	Upscaled from 32×32
CIFAR10 (Krizhevsky et al., 2009)	50,000	10,000	10	128	Upscaled from 32×32
SVHN (Netzer et al., 2011)	73,257	26,032	10	128	Upscaled from 32×32
Stanford Dogs (Khosla et al., 2011)	11,999	8,580	120	224	
VGG Pets (Parkhi et al., 2012)	3,680	3,669	37	224	
VGG Flowers (Nilsback & Zisserman, 2008)	2,040	6,149	102	224	
Food-101 (Bossard et al., 2014)	75,750	25,250	101	224	
CUB (Wah et al., 2011)	5,994	5,794	200	224	
DTD (Cimpoi et al., 2014)	4,230	1,410	47	224	Texture dataset

C.4. Empirical error analysis

We explain the experimental setup for **the Figure 3 in the main paper**. We define the approximation error as:

$$\varepsilon := U_k(\phi + \Delta_1 + \cdots + \Delta_{k-1}) - U_1(\cdots U_1(U_1(\phi) + \Delta_1) \cdots + \Delta_{k-1}) \tag{2}$$

and report $\log_{10} \|\varepsilon\|_2$ versus inner-learning rate α, meta-learning rate β, length of inner-learning trajectory k, and type of network activations (ReLU vs. Softplus).

- We use ResNet20.
- We use the first split of TinyImageNet dataset to generate a sequence of losses $\mathcal{L}_0, \ldots, \mathcal{L}_{k-1}$. Batch size is set to 128.
- We use vanilla SGD for U_k, such that $U_k(\phi) := \phi - \alpha \sum_{i=0}^{k-1} \nabla_\theta \mathcal{L}_i|_{\theta=\theta_i}$.
- Note that $\Delta_i = \beta \cdot \mathsf{MetaGrad}_i$. In order to compute the approximation error ε in a feasible amount of time, we assume that $\mathsf{MetaGrad}_0, \ldots, \mathsf{MetaGrad}_{k-1}$ are sampled from the following synthetic distribution: $\mathsf{MetaGrad}_i = x_i / \|x_i\|_2$, where $x_i \sim N(0, I)$ for $i = 0, \ldots, k-1$. Therefore, we have $\|\Delta_0\|_2 = \cdots = \|\Delta_{k-1}\|_2 = \beta$, i.e. the size of meta-update is fixed as β, but the direction is randomized. We use the same sequence of $\Delta_0, \ldots, \Delta_{k-1}$ in computing the first and second term of Eq. (2).

D. Derivation of the Error Complexity

In this section, we derive **Eq. (5) in the main paper**, the complexity of the approximation error. We first recap the notations:

- ϕ: Shared initial model parameters that we meta-learn
- α: Inner-learning rate
- β: Meta-learning rate
- $U_k(\omega)$: Task-specific parameters after k SGD steps from ω, i.e.

$$U_k(\omega) = U_{k-1}(\omega) - \alpha \nabla_\omega \mathcal{L}_{k-1}|_{\omega=U_{k-1}(\omega)} = \omega - \alpha \sum_{i=0}^{k-1} \nabla_\omega \mathcal{L}_i$$

- θ_k: Task-specific parameters after k SGD steps from ϕ, i.e. $U_k(\phi)$
- H_k: Hessian of loss function at θ_k
- Δ_k: Meta-update (or trajectory shifting) at step k, i.e.

$$\Delta_k = -\beta \cdot \mathsf{MetaGrad}(\phi; \theta_k)$$

Our derivation is based on the following assumptions:

1. U_k is infinitely differentiable.

2. Norm of the Hessian is bounded by h at everywhere, i.e. $\|H\| = O(h)$.

3. Norm of meta-update is bounded by β for every step, i.e. $\|\Delta_k\| = O(\beta)$ for every k.

Theorem D.1. *For $k \geq 1$ and any Δ whose norm is sufficiently small,*

$$U_k(\phi + \Delta) = U_k(\phi) + \Delta + O(\alpha h k \|\Delta\| + \|\Delta\|^2)$$

Proof. Using the Talyor approximation,

$$\begin{aligned} U_k(\phi + \Delta) &= U_k(\phi) + \frac{\partial U_k(\phi)}{\partial \phi}\Delta + \frac{1}{2}\Delta^\top \frac{\partial^2 U_k(\phi)}{\partial \phi^2}\Delta + \cdots \\ &= U_k(\phi) + \frac{\partial U_k(\phi)}{\partial \phi}\Delta + O(\|\Delta\|^2) \end{aligned}$$

On the other hand,

$$\begin{aligned} \frac{\partial U_k(\phi)}{\partial \phi} &= \frac{\partial U_k(\phi)}{\partial U_{k-1}(\phi)} \cdots \frac{\partial U_1(\phi)}{\partial \phi} = \prod_{i=0}^{k-1} (I - \alpha H_i) \\ &= I - \sum_{i=0}^{k-1} \alpha H_i + \sum_{i=0}^{k-1} \sum_{j=i+1}^{k-1} \alpha^2 H_i H_j + \cdots = I + O(\alpha h k) \end{aligned}$$

Combining the two approximations,

$$\begin{aligned} U_k(\phi + \Delta) &= U_k(\phi) + (I + O(\alpha h k))\Delta + O(\|\Delta\|^2) \\ &= U_k(\phi) + \Delta + O(\alpha h k \|\Delta\| + \|\Delta\|^2) \end{aligned}$$

□

Theorem D.2. *For $k \geq 1$ and any $\{\Delta_i\}$ with $\|\Delta_i\| = O(\beta)$,*

$$U_k\left(\phi + \sum_{i=1}^{k} \Delta_i\right) = U_1(\cdots U_1(U_1(\phi) + \Delta_1) \cdots + \Delta_{k-1}) + \Delta_k + O(\beta \alpha h k^2 + \beta^2 k)$$

Proof. For $k = 1$,

$$U_1(\phi + \Delta_1) = U_1(\phi) + \Delta_1 + O(\beta \alpha h + \beta^2) \qquad \text{(Theorem D.1)}$$

With the assumption at step k,

$$U_{k+1}\left(\phi+\sum_{i=1}^{k+1}\Delta_i\right)=U_{k+1}\left(\phi+\sum_{i=1}^{k}\Delta_i\right)+\Delta_{k+1}+O\left(\alpha h(k+1)\|\Delta_{k+1}\|+\|\Delta_{k+1}\|^2\right)$$

(Theorem D.1)

$$\begin{aligned}
&= U_1\left(U_k\left(\phi+\sum_{i=1}^{k}\Delta_i\right)\right)+\Delta_{k+1}+O(\beta\alpha h(k+1)+\beta^2)\\
&= U_1\left(U_1\left(\cdots U_1(U_1(\phi)+\Delta_1)\cdots+\Delta_{k-1}\right)+\Delta_k+O(\beta\alpha hk^2+\beta^2k)\right)\\
&\quad+\Delta_{k+1}+O(\beta\alpha h(k+1)+\beta^2)
\end{aligned}$$

(Assumption at step k)

$$\begin{aligned}
&= U_1\left(U_1(\cdots U_1(U_1(\phi)+\Delta_1)\cdots+\Delta_{k-1})+\Delta_k\right)+O(\beta\alpha hk^2+\beta^2k)\\
&\quad+O(\alpha h(\beta\alpha hk^2+\beta^2k)+(\beta\alpha hk^2+\beta^2k)^2)\\
&\quad+\Delta_{k+1}+O(\beta\alpha h(k+1)+\beta^2)
\end{aligned}$$

(Theorem D.1)

$$= U_1(U_1(\cdots U_1(U_1(\phi)+\Delta_1)\cdots+\Delta_{k-1})+\Delta_k)+\Delta_{k+1}+O(\beta\alpha h(k+1)^2+\beta^2(k+1))$$

□

Corollary 1. *Asymptotic approximation error of proposed **continual trajectory shifting** is as follows:*

$$U_k\left(\phi+\sum_{i=1}^{k-1}\Delta_i\right)=U_1(\cdots U_1(U_1(\phi)+\Delta_1)\cdots+\Delta_{k-1})+O(\beta\alpha hk^2+\beta^2k)$$

for $k\geq 2$.

Proof.

$$\begin{aligned}
U_k\left(\phi+\sum_{i=1}^{k-1}\Delta_i\right)&=U_1\left(U_{k-1}\left(\phi+\sum_{i=1}^{k-1}\Delta_i\right)\right)\\
&=U_1(U_1(\cdots U_1(U_1(\phi)+\Delta_1)\cdots)+\Delta_{k-1}+O(\beta\alpha h(k-1)^2+\beta^2(k-1)))
\end{aligned}$$

(Theorem D.2)

$$\begin{aligned}
&=U_1(\cdots U_1(U_1(\phi)+\Delta_1)\cdots+\Delta_{k-1})+O(\beta\alpha h(k-1)^2+\beta^2(k-1))\\
&\quad+O(\alpha h(\beta\alpha h(k-1)^2+\beta^2(k-1))+(\beta\alpha h(k-1)^2+\beta^2(k-1))^2)
\end{aligned}$$

(Theorem D.1)

$$=U_1(\cdots U_1(U_1(\phi)+\Delta_1)\cdots+\Delta_{k-1})+O(\beta\alpha hk^2+\beta^2k)$$

□

E. Derivation for the momentum optimizer

In this section, we prove that we can use the same shifting rule even with the momentum optimizer and weight decaying. See Section A for more discussion about the empirical effect of the type of inner-optimizer.

Momentum Note that the update function of SGD with momentum μ is:

$$\begin{aligned}
U_k(\omega)&=U_{k-1}(\omega)-\alpha\cdot g_k(\omega)\\
\text{where } g_k(\omega)&=\mu\cdot g_{k-1}(\omega)+\nabla_\omega\mathcal{L}_{k-1}|_{\omega=U_{k-1}(\omega)}.
\end{aligned}$$

Then, the following lemma holds:

Lemma E.1. *The approximation of Jacobian is*

$$\frac{\partial U_k(\phi)}{\partial \phi} = I + O\left(\frac{\alpha h k}{1-\mu}\right)$$

Proof. Let the approximation error of Jacobian at step k be ϵ_k, i.e.

$$\frac{\partial U_k(\phi)}{\partial \phi} = I + \epsilon_k$$

For $k \geq 2$,

$$\begin{aligned}
U_k(\phi) &= U_{k-1}(\phi) - \alpha \cdot g_k(\phi) \\
&= U_{k-1}(\phi) - \alpha \cdot (\mu \cdot g_{k-1}(\phi) + \nabla_\phi \mathcal{L}_{k-1}|_{\phi=U_{k-1}(\phi)}) \\
&= U_{k-1}(\phi) - \mu \cdot \alpha \cdot g_{k-1}(\phi) - \alpha \cdot \nabla_\phi \mathcal{L}_{k-1}|_{\phi=U_{k-1}(\phi)} \\
&= U_{k-1}(\phi) - \mu \cdot (U_{k-1}(\phi) - U_{k-2}(\phi)) - \alpha \cdot \nabla_\phi \mathcal{L}_{k-1}|_{\phi=U_{k-1}(\phi)}
\end{aligned}$$

Then, we compute the Jacobian as

$$\begin{aligned}
\frac{\partial U_k(\phi)}{\partial \phi} &= \frac{\partial U_{k-1}(\phi)}{\partial \phi} - \mu \cdot \left(\frac{\partial U_{k-1}(\phi)}{\partial \phi} - \frac{\partial U_{k-2}(\phi)}{\partial \phi}\right) - \alpha \cdot \frac{\partial \nabla_\phi \mathcal{L}_{k-1}|_{\phi=U_{k-1}(\phi)}}{\partial \phi} \\
&= (I + \epsilon_{k-1}) + \mu \cdot (\epsilon_{k-1} - \epsilon_{k-2}) - \alpha \cdot \frac{\partial \nabla_\phi \mathcal{L}_{k-1}|_{\phi=U_{k-1}(\phi)}}{\partial U_{k-1}(\phi)} \cdot \frac{\partial U_{k-1}(\phi)}{\partial \phi} \\
&= I + \epsilon_{k-1} + \mu \cdot (\epsilon_{k-1} - \epsilon_{k-2}) - \alpha H_{k-1} \cdot (I + \epsilon_{k-1})
\end{aligned}$$

Thus,

$$\epsilon_k = \epsilon_{k-1} + \mu \cdot (\epsilon_{k-1} - \epsilon_{k-2}) + O(\alpha h(1 + \epsilon_{k-1}))$$

We can say $\epsilon_0 = 0$ and for $k = 1$,

$$\frac{\partial U_1(\phi)}{\partial \phi} = \frac{\partial(\phi - \alpha \cdot \nabla_\phi \mathcal{L}_0|_{\phi=\phi})}{\partial \phi} = I - \alpha H_0 \qquad \rightarrow \qquad \epsilon_1 = O(\alpha h)$$

Then,

$$\begin{aligned}
\epsilon_k - \epsilon_{k-1} &= \mu \cdot (\epsilon_{k-1} - \epsilon_{k-2}) + O(\alpha h(1 + \epsilon_{k-1})) \\
&= \mu^2 \cdot (\epsilon_{k-2} - \epsilon_{k-3}) + O(\alpha h(1 + \epsilon_{k-1} + \mu(1 + \epsilon_{k-2}))) \\
&= \cdots \\
&= \mu^{k-1} \cdot (\epsilon_1 - \epsilon_0) + O\left(\alpha h \left(\sum_{i=0}^{k-2} \mu^i (1 + \epsilon_{k-1-i})\right)\right) \\
&= O\left(\alpha h \left(\sum_{i=0}^{k-1} \mu^i + \sum_{i=0}^{k-2} \mu^i \epsilon_{k-1-i})\right)\right)
\end{aligned}$$

Since the second term $O(\alpha h \sum_{i=0}^{k-2} \mu^i \epsilon_{k-1-i})$ is quadratic to αh, dropping the term,

$$\begin{aligned}
\epsilon_k &= \epsilon_{k-1} + O\left(\alpha h \sum_{i=0}^{k-1} \mu^i\right) \\
&= \epsilon_{k-1} + O\left(\alpha h \frac{1-\mu^k}{1-\mu}\right) \\
&= \epsilon_{k-2} + O\left(\alpha h \left(\frac{1-\mu^k}{1-\mu} + \frac{1-\mu^{k-1}}{1-\mu}\right)\right) \\
&= \cdots \\
&= \epsilon_0 + O\left(\alpha h \sum_{i=1}^{k} \frac{1-\mu^i}{1-\mu}\right) \\
&= O\left(\frac{\alpha h}{1-\mu}\left(k + 1 - \frac{1-\mu^{k+1}}{1-\mu}\right)\right) = O\left(\frac{\alpha h k}{1-\mu}\right)
\end{aligned}$$

□

Then, as Theorem D.1, approximation error between $U_k(\phi + \Delta)$ and $U_k(\phi) + \Delta$ is as follows:

Theorem E.1. *For any Δ that with sufficiently small norm,*

$$U_k(\phi + \Delta) = U_k(\phi) + \Delta + O\left(\frac{\alpha h k}{1-\mu}\|\Delta\| + \|\Delta\|^2\right)$$

and as Corollary 1,

Corollary 2. *Asymptotic approximation error of proposed **Continual Correction** with SGD and momentum μ is as follows:*

$$U_k\left(\phi + \sum_{i=1}^{k-1} \Delta_i\right) = U_1(\cdots U_1(U_1(\phi) + \Delta_1) \cdots + \Delta_{k-1}) + O\left(\frac{\beta \alpha h k^2}{1-\mu} + \beta^2 k\right)$$

for $k \geq 2$.

We omit the proofs since only the coefficients are different.

Weight decay Denote the update function of SGD with weight decay λ by:

$$U_k(\omega) = U_{k-1}(\omega) - \alpha \cdot \left(\nabla_\omega \left(\mathcal{L}_{k-1}|_{\omega = U_{k-1}(\omega)} + \lambda \|U_{k-1}(\omega)\|^2\right)\right).$$

Then, following lemma holds:

Lemma E.2. *Approximation of Jacobian is*

$$\frac{\partial U_k(\phi)}{\partial \phi} = I + O\left(\alpha(h + 2\lambda)k\right)$$

Proof.

$$\begin{aligned}
\frac{\partial U_k(\phi)}{\partial \phi} &= \frac{\partial U_k(\phi)}{\partial U_{k-1}(\phi)} \cdots \frac{\partial U_1(\phi)}{\partial \phi} = \prod_{i=0}^{k-1} (I - \alpha H_i + 2\lambda) \\
&= I + O(\alpha(h + 2\lambda)k)
\end{aligned}$$

□

Then, as previous,

Corollary 3. *Asymptotic approximation error of proposed* ***continual trajectory shifting*** *with SGD and weight decay λ is as follows:*

$$U_k\left(\phi+\sum_{i=1}^{k-1}\Delta_i\right)=U_1(\cdots U_1(U_1(\phi)+\Delta_1)\cdots+\Delta_{k-1})+O\left(\beta\alpha(h+2\lambda)k^2+\beta^2k\right)$$

for $k\geq 2$.

References

https://tiny-imagenet.herokuapp.com/.

Bossard, L., Guillaumin, M., and Gool, L. V. Food-101 - mining discriminative components with random forests. In *ECCV*, 2014.

Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., , and Vedaldi, A. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Coates, A., Ng, A., and Lee, H. An Analysis of Single-Layer Networks in Unsupervised Feature Learning. In *AISTATS*, 2011.

Finn, C., Abbeel, P., and Levine, S. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *ICML*, 2017.

Flennerhag, S., Moreno, P. G., Lawrence, N., and Damianou, A. Transferring Knowledge across Learning Processes. In *ICLR*, 2019.

Ha, D. and Eck, D. A neural representation of sketch drawings. *CoRR*, abs/1704.03477, 2017. URL http://arxiv.org/abs/1704.03477.

Khosla, A., Jayadevaprakash, N., Yao, B., and Fei-Fei, L. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, CVPR*, 2011.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, 2013.

Krizhevsky, A., Hinton, G., et al. Learning Multiple Layers of features from Tiny Images. 2009.

Maji, S., Rahtu, E., Kannala, J., Blaschko, M., and Vedaldi, A. Fine-Grained Visual Classification of Aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading Digits in Natural Images with Unsupervised Feature Learning. 2011.

Nichol, A., Achiam, J., and Schulman, J. On First-Order Meta-Learning Algorithms. *arXiv e-prints*, 2018.

Nilsback, M.-E. and Zisserman, A. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

Parkhi, O. M., Vedaldi, A., Zisserman, A., and Jawahar, C. V. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. On the importance of initialization and momentum in deep learning. In *ICML*, 2013.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*, 2017.