DeepZero: Scaling up Zeroth-Order Optimization for Deep Model Training

Aochuan Chen*[1], Yimeng Zhang*[1],
Jinghan Jia[1], James Diffenderfer[2], Jiancheng Liu[1], Konstantinos Parasyris[2], Yihua Zhang[1], Zheng Zhang[3],
Bhavya Kailkhura[2], Sijia Liu[1]
[1]Michigan State University, [2]Lawrence Livermore National Laboratory, [3]UC Santa Barbara

Scan For Code

Zeroth-Order Optimization: How to optimize without first-order gradient?

- by finite-difference: $\hat{\nabla}_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}) = \frac{1}{q}\sum_{i=1}^{q}\left[\frac{\ell(\boldsymbol{\theta}+\mu\mathbf{u}_i)-\ell(\boldsymbol{\theta})}{\mu}\mathbf{u}_i\right]$
- to tackle with black-box involved optimization, such as non-differentiable physical simulators[1].
- to unleash the potential of special hardware, such as ONNs[2] (optical neural networks).

❖ Gradient Estimator: CGE over RGE

- ➢ CGE uses basis vectors while RGE uses random vectors.

$$(\text{RGE})\ \hat{\nabla}_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}) = \frac{1}{q}\sum_{i=1}^{q}\left[\frac{\ell(\boldsymbol{\theta}+\mu\mathbf{u}_i)-\ell(\boldsymbol{\theta})}{\mu}\mathbf{u}_i\right];\ (\text{CGE})\ \hat{\nabla}_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}) = \sum_{i=1}^{d}\left[\frac{\ell(\boldsymbol{\theta}+\mu\mathbf{e}_i)-\ell(\boldsymbol{\theta})}{\mu}\mathbf{e}_i\right]$$

- ➢ Experiments show that CGE outperforms RGE in terms of accuracy given the same query budget
- ➢ Runtime profiling exhibits CGE's efficiency merit over RGE at the same number of queries

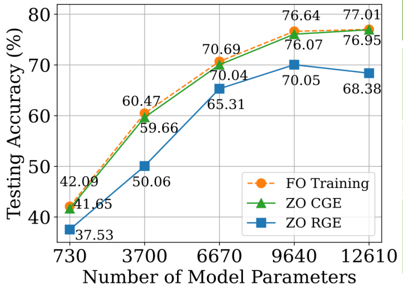

Figure 1. Performance comparison on a simple CNN with varying numbers of parameters on CIFAR-10.

	CGE	RGE
Accuracy (even q = d)	😃	
Computation efficiency (q = d)	😃	
Query efficiency (q < d)		😃

Table 1. Comparison between CGE and RGE from the perspective of accuracy, computation, and query efficiency.

Reference [1] Kiwon Um et al., Solver-in-the-loop: Learning from differentiable physics to interact with iterative pde-solvers. [2] Jiaqi Gu, et al., L2ight: Enabling on-chip learning for optical neural networks via efficient in-situ subspace optimization. [3] Chaoqi Wang et al, Picking winning tickets before training by preserving gradient flow.

❖ DeepZero: Sparse Gradients Guided by ZO Pruning

- ➢ The disentanglement of weights within CGE is inherently pruning-friendly.
- ➢ We extend a pruning-at-initialization method, GraSP [3], to its ZO version:

$$\hat{\mathbf{S}} := -\boldsymbol{\theta} \odot \frac{\hat{\nabla}_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}+\mu\hat{\mathbf{g}}) - \hat{\nabla}_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta})}{\mu}$$

- ➢ Guided by ZO-GraSP, we then introduce **dynamic sparse pattern**, leading to sparse ZO gradient estimates

Algorithm 1 ZO-GraSP-Guided ZO Training

1: Get $\mathcal{S}_{\text{ZO-GraSP}}$ through ZO-GraSP (3)
2: Obtain layer-wise pruning ratio $\mathcal{S}_{\text{layer}}$ based on $\mathcal{S}_{\text{ZO-GraSP}}$
3: **for** Epoch $t = 0, 1, 2, \ldots, T-1$ **do**
4: Randomly generate a sparse coordinate set $\mathcal{S}_t$ according to $\mathcal{S}_{\text{layer}}$
5: **for** Iterations per epoch **do**
6: Obtain (Sparse-CGE) $\hat{\nabla}_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta})$ based on $\mathcal{S}_t$
7: Update model weights: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha\hat{\nabla}_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta})$
8: **end for**
9: **end for**

❖ Acceleration: Feature Reuse & Forward Parallel

Feature Reuse: $f_{\boldsymbol{\theta}}(\mathbf{x}) = f_{\boldsymbol{\theta}_{>l}}(\mathbf{z}_l) = \underbrace{f_{\boldsymbol{\theta}_L} \circ f_{\boldsymbol{\theta}_{L-1}} \circ \cdots \circ f_{\boldsymbol{\theta}_{l+1}}}_{f_{\boldsymbol{\theta}_{>l}}(\cdot)} \circ \underbrace{f_{\boldsymbol{\theta}_l} \circ \cdots \circ f_{\boldsymbol{\theta}_1}(\mathbf{x})}_{\mathbf{z}_l = f_{\boldsymbol{\theta}_{1:l}}(\mathbf{x})},$

Forward Parallel: $\hat{\nabla}_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}) = \sum_{i=1}^{M}\hat{\mathbf{g}}_i,\ \hat{\mathbf{g}}_i := \sum_{j\in\mathcal{S}_i}\left[\frac{\ell(\boldsymbol{\theta}+\mu\mathbf{e}_j)-\ell(\boldsymbol{\theta})}{\mu}\mathbf{e}_j\right]$

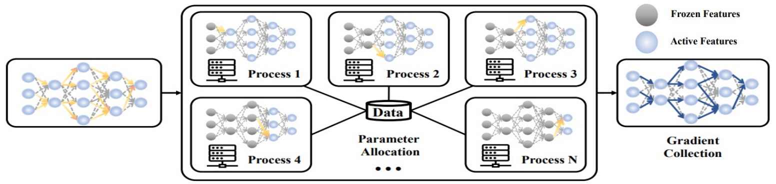

Figure 2. Forward Parallel: Queries are totally independent of each other and thus can be easily parallelized without any performance loss. Feature Reuse: one can reuse the feature immediately preceding the perturbed layer thanks to the minimum coordinate-wise perturbation by CGE.

❖ Experiments and Applications

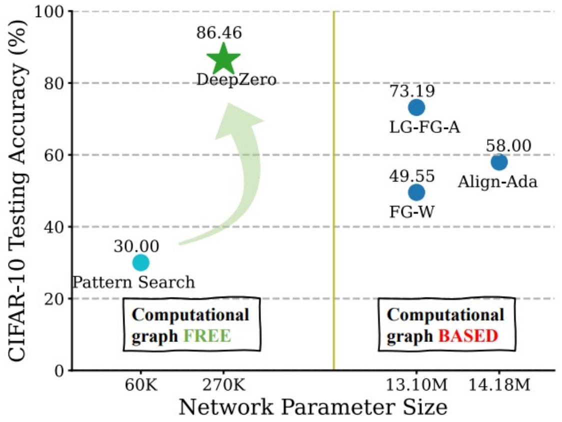

Figure 3. DeepZero outperforms the computational-graph-free baseline Pattern Search and computational-graph-dependent non-BP methods (ResNet-20, CIFAR10).

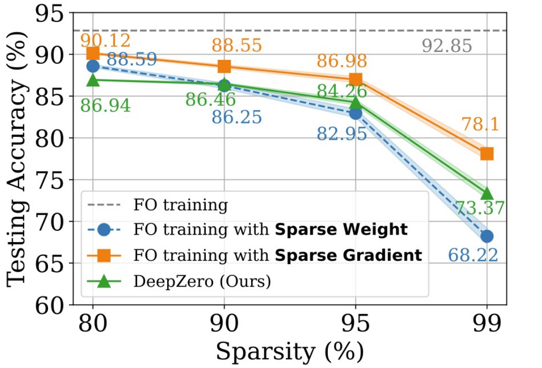

Figure 4. Comparison between DeepZero and FO training baselines on a ResNet-20 for CIFAR-10.

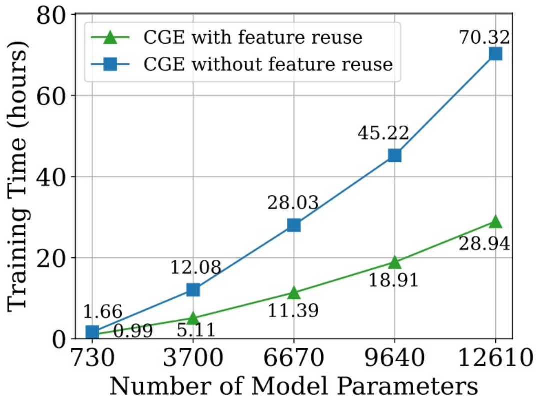

Figure 5. Computation cost of CGE-based ZO training w/ Feature Reuse vs. w/o Feature Reuse. Empirically, Feature Reuse can half the running time.

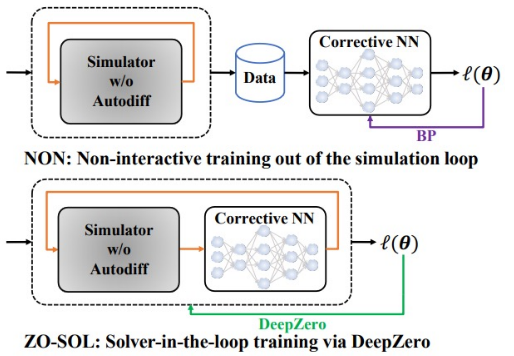

Figure 6. In physical simulation-involved tasks, DeepZero enables 'solver-in-the-loop' training for non-differentiable simulators.

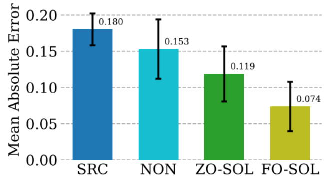

Figure 7. DeepZero approaches the first-order 'solver-in-the-loop'. Performance measured by mean absolute error (MAE, lower better).