Differential Dynamic Programming Neural Optimizer

Guan-Horng Liu

with Tianrong Chen, and Evangelos A. Theodorou
Georgia Institute of Technology

{*ghliu,tianrong.chen,evangelos.theodorou*}*@gatech.edu*

ICLR 2021, May

Introduction

- Dynamical system perspective of DNNs has received great attentions.
  - e.g. ResNet propagation rule, $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + f_\theta(\boldsymbol{x}_k)$, as a forward-Euler discretization of an ODE system $\dot{\boldsymbol{x}}_k = f_\theta(\boldsymbol{x}_k)$.
- Despite enahncing theoretical understanding of DNNs and inspiring new architectures from numerical differential equations [Weinan(2017)] [Greydanus et al.(2019), Liu & Theodorou(2019), Chen et al.(2018)], the algorithmic development remains limited.

Motivation

Can we propose new class of training optimizer, inspired from Optimal Control theory, that is principled from the dynamical system viewpoint while being scalable to high-dimensional ML problems?

Training DNNs by Solving Optimal Control

- Problem formulation of Optimal Control Programming (OCP):

$$\min_{\bar{\boldsymbol{u}} \triangleq \{\boldsymbol{u}_t\}_{t=0}^{T-1}} J(\bar{\boldsymbol{u}}; \boldsymbol{x}_0) := \left[\phi(\boldsymbol{x}_T) + \sum_{t=0}^{T-1} \ell_t(\boldsymbol{x}_t, \boldsymbol{u}_t) \right] \text{,s.t. } \boldsymbol{x}_{t+1} = f_t(\boldsymbol{x}_t, \boldsymbol{u}_t). \quad (1)$$

- OCP describes the optimization process of DNN training.

Table: Terminology mapping.

	Optimal Control	DNN Training
J	Trajectory Cost	Total Loss
t	Discrete Time Step	Layer Index
$\boldsymbol{x}$	State Vector	Vectorized Activation
$\boldsymbol{u}$	Control Vector	Weight Parameter
f	Dynamical System	Layer Propagation
ϕ	Terminal Cost	End-goal Loss
ℓ	Intermediate Cost	Weight Decay

Bellman Optimality

Theorem 1 (Dynamic Programming Principle)

Define a value function V_t at each time step that is computed backward in time using the Bellman equation

$$V_t(\boldsymbol{x}_t) = \min_{\boldsymbol{u}_t(\boldsymbol{x}_t)\in\Gamma_{\boldsymbol{x}_t}} \underbrace{\ell_t(\boldsymbol{x}_t, \boldsymbol{u}_t) + V_{t+1}(f_t(\boldsymbol{x}_t, \boldsymbol{u}_t))}_{Q_t(\boldsymbol{x}_t, \boldsymbol{u}_t)\equiv Q_t}, \quad V_T(\boldsymbol{x}_T) = \phi(\boldsymbol{x}_T), \tag{1}$$

where $\Gamma_{\boldsymbol{x}_t}$ denotes a set of mapping from state to control space. Then, we have $V_0(\boldsymbol{x}_0) = J^*(\boldsymbol{x}_0)$ be the optimal objective value to OCP. Further, let $\boldsymbol{u}_t^*(\boldsymbol{x}_t)$ be the minimizer of (1) for each t, then $\{\boldsymbol{u}_t^*(\boldsymbol{x}_t)\}_{t=0}^{T-1}$ is globally optimal policy.

Differential Dynamic Programming Neural Optimizer

- DDPNOpt solves the Bellman Eq (2) iteratively and approximately.
- At each iteration and for a given nominal trajectory $(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})$, we solve

$$\tilde{V}_t = \min_{\delta \boldsymbol{u}_t(\delta \boldsymbol{x}_t) \in \Gamma'_{\delta x_t}} \frac{1}{2} \begin{bmatrix} \mathbf{1} \\ \delta \boldsymbol{x}_t \\ \delta \boldsymbol{u}_t \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \mathbf{0} & {Q^t_{\boldsymbol{x}}}^{\mathsf{T}} & {Q^t_{\boldsymbol{u}}}^{\mathsf{T}} \\ Q^t_{\boldsymbol{x}} & Q^t_{\boldsymbol{xx}} & Q^t_{\boldsymbol{xu}} \\ Q^t_{\boldsymbol{u}} & Q^t_{\boldsymbol{ux}} & Q^t_{\boldsymbol{uu}} \end{bmatrix} \begin{bmatrix} \mathbf{1} \\ \delta \boldsymbol{x}_t \\ \delta \boldsymbol{u}_t \end{bmatrix}, \qquad (2)$$

where $Q^t_{\boldsymbol{x}} \equiv \nabla_{\boldsymbol{x}} Q_t$, $Q^t_{\boldsymbol{xx}} \equiv \nabla^2_{\boldsymbol{x}} Q_t$, etc.

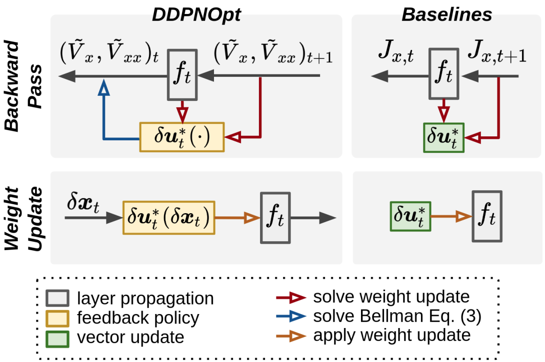
- Analytic solution to (2) is given by a linear *feedback policy*.

$$\delta \boldsymbol{u}^*_t(\delta \boldsymbol{x}_t) = \boldsymbol{k}_t + \boldsymbol{K}_t \delta \boldsymbol{x}_t \ ,$$
$$\text{where } \boldsymbol{k}_t \triangleq -(Q^t_{\boldsymbol{uu}})^{-1} Q^t_{\boldsymbol{u}} \text{ and } \boldsymbol{K}_t \triangleq -(Q^t_{\boldsymbol{uu}})^{-1} Q^t_{\boldsymbol{ux}}. \qquad (3)$$

- These precedures (3-4) repeat recursively backward from T to $t = 0$.

Algorithmic Similarity

- Computation graph of backward pass and weight update.



Connection with Baselines

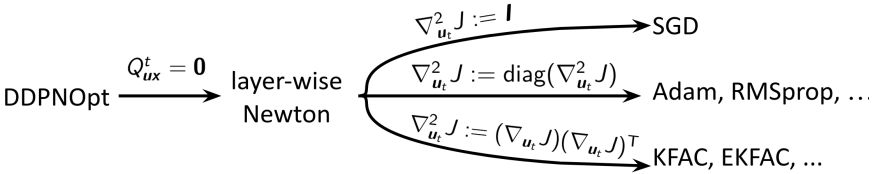
Theorem 2

Assume $Q^t_{\boldsymbol{ux}} = \mathbf{0}$ *at all stages, then we have* $V_{\boldsymbol{x},t} = J_{\boldsymbol{x},t}, \forall t$*. In this case, DDPNOpt is equivalent to stage-wise Newton, where the gradient is preconditioned by the block-wise inverse Hessian at each layer:*

$$\delta \boldsymbol{u}^*_t(\delta \boldsymbol{x}_t) = \boldsymbol{k}_t + \boldsymbol{K}_t \delta \boldsymbol{x}_t = -(\nabla^2_{\boldsymbol{u}_t} J)^{-1} \nabla_{\boldsymbol{u}_t} J \ . \tag{4}$$

If further we have $Q^t_{\boldsymbol{uu}} := \boldsymbol{I}$*, then DDPNOpt degenerates to SGD.*

- Most 1st and 2nd order methods are special cases of DDPNOpt.



Connection with Baselines

Theorem 2

Assume $Q^t_{\boldsymbol{ux}} = \mathbf{0}$ at all stages, then we have $V_{\boldsymbol{x},t} = J_{\boldsymbol{x},t}, \forall t$. In this case, DDPNOpt is equivalent to stage-wise Newton, where the gradient is preconditioned by the block-wise inverse Hessian at each layer:

$$\delta \boldsymbol{u}^*_t(\delta \boldsymbol{x}_t) = \boldsymbol{k}_t + \boldsymbol{K}_t \delta \boldsymbol{x}_t = -(\nabla^2_{\boldsymbol{u}_t} J)^{-1} \nabla_{\boldsymbol{u}_t} J \ . \tag{5}$$

If further we have $Q^t_{\boldsymbol{uu}} := \boldsymbol{I}$, then DDPNOpt degenerates to SGD.

- Most 1st and 2nd order methods are special cases of DDPNOpt.

SGD

All baselines can be extended to accept Bellman Optimality
by incorporating non-trivial mixed derivatives ($\nabla_{\boldsymbol{ux}}$)

DDPNOpt $\longleftarrow$ Adam, RMSprop, …

KFAC, EKFAC, ...

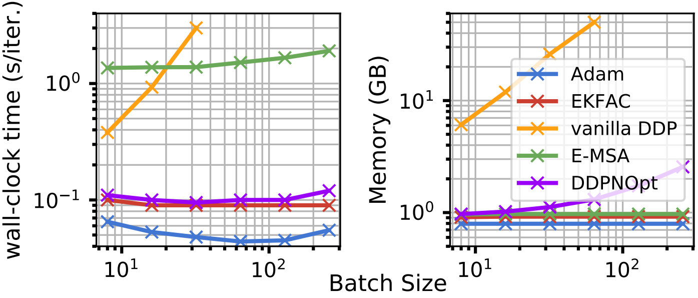
Experiment Result: Classification Datasets

- Test accuracy (%) averaged over 10 seeds.
  - DDPNOpt improves test accuracy for both CNN and feedforward networks, and outperform other optimal-contorl inspired baselines.

	DataSet	Standard baselines SGD-m	RMSProp	Adam	EKFAC	OCP-inspired baselines E-MSA	vanilla DDP	**DDPNOpt (ours)**
Feed-forward	WINE	94.35	98.10	98.13	94.60	93.56	98.00	**98.18**
	DIGITS	**95.36**	94.33	94.98	95.24	94.87	91.68	95.13
	MNIST	92.65	91.89	92.54	92.73	90.24	90.42	**93.30**
	F-MNIST	82.49	83.87	84.36	84.12	82.04	81.98	**84.98**
CNN	MNIST	97.94	98.05	98.04	98.02	96.48	N/A	**98.09**
	SVHN	89.00	88.41	87.76	90.63	79.45		**90.70**
	CIFAR-10	71.26	70.52	70.04	71.85	61.42		**71.92**

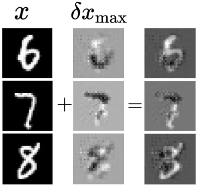
Experiment Result: Computational Complexity

- **Baselines**:
  - Standard: Adam (1st-order), EKFAC (2nd-order)
  - OCP-inspired: E-MSA (2st-order), vanilla DDP (2nd-order)

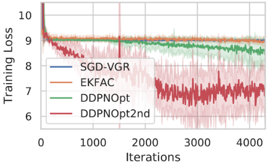


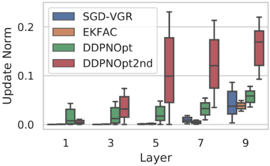
Experiment Result: Effect of Feedback Policies

- Visualization of on MNIST suggests that the feedback policy captures non-trivial visual features ($\delta \boldsymbol{x}_{\max}$) related to the pixel-wise difference between spatially similar classes.
- The feedback policy can help mitigate vanishing gradient for sigmoid-activated networks during Back-propagation.



Vanishing Gradient Results





Conclusion

- DDP Neural Optimizer is a new class of training method that inherits Bellman optimality from Optimal Control Theory. It generalizes most existing baselines as special cases and generates layer-wise feedback updates that improve overall training performance and numerical stabilities.

Paper **Poster**

References

Guan-Horng Liu, Tianrong Chen, and Evangelos A. Theodorou
Differential Dynamic Programming Neural Optimizer
arXiv preprint arXiv:2002.08809, 2020.

E Weinan.
A proposal on machine learning via dynamical systems.
Communications in Mathematics and Statistics, 5(1):1–11, 2017.

Guan-Horng Liu and Evangelos A Theodorou.
Deep learning theory review: An optimal control and dynamical systems perspective.
arXiv preprint arXiv:1908.10920, 2019.

Guan-Horng Liu, Tianrong Chen, and Evangelos A. Theodorou
A Differential Game Theoretic Neural Optimizer for Training Residual Networks
arXiv preprint arXiv:2007.08880, 2020.

Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong.
Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations.
arXiv preprint arXiv:1710.10121, 2017.

Samuel Greydanus, Misko Dzamba, and Jason Yosinski.
Hamiltonian neural networks.
In *Advances in Neural Information Processing Systems*, pp. 15353–15363, 2019.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud.
Neural ordinary differential equations.
In *Advances in Neural Information Processing Systems*, pp. 6572–6583, 2018.

Lev Semenovich Pontryagin, EF Mishchenko, VG Boltyanskii, and RV Gamkrelidze.
The mathematical theory of optimal processes.