What does automatic differentiation compute for neural networks?

Sejun Park[1*], Sanghyuk Chun[2*], Wonyeol Lee[3]

[1]Korea University, [2]NAVER AI Lab, [3]Carnegie Mellon University

TL;DR. We theoretically study when AD computes a correct derivative. We empirically show that practical NNs satisfy our sufficient conditions for the correctness.

Summary

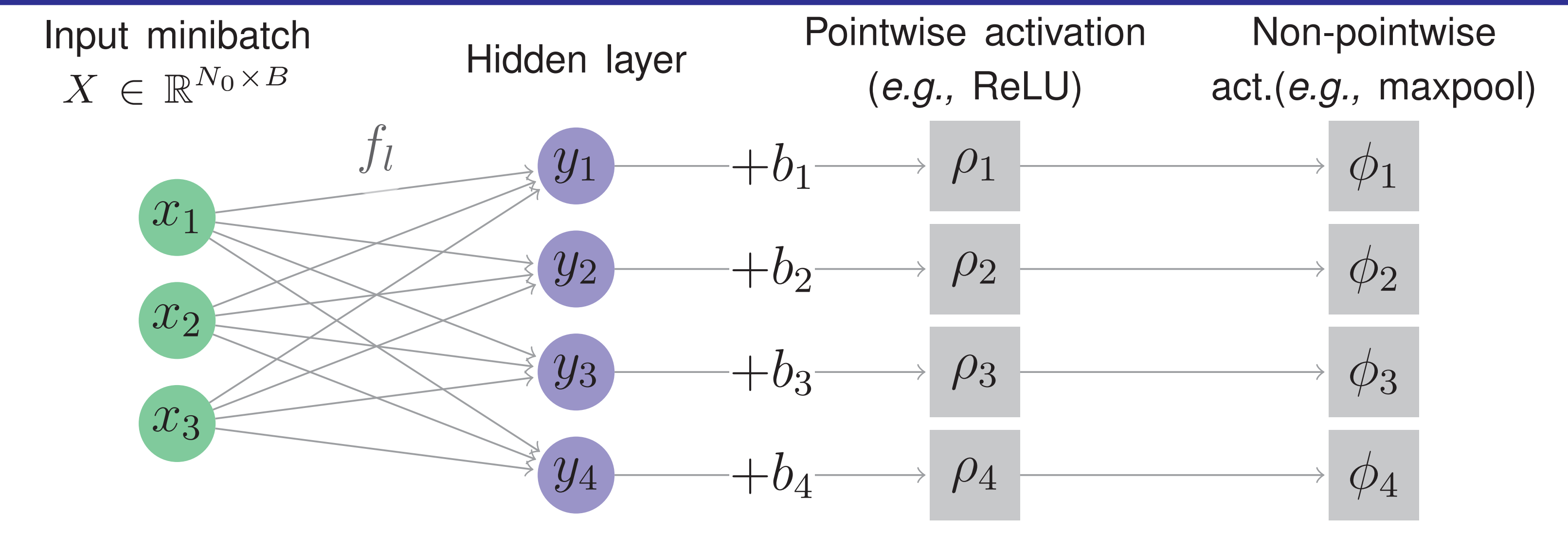
- Automatic differentiation (AD) is an efficient, popular algorithm to compute the derivative of neural networks (NNs). It is based on the chain rule, and consists of the forward mode and reverse mode.
- However, if a NN uses non-differentiable functions (e.g., ReLU), AD might not compute a correct derivative, (i.e., a generalized derivative called a Clarke subderivative). Prior works have shown that AD is correct under specific conditions.
- This work shows that AD is always correct for NNs with pointwise activation functions (e.g., ReLUs), if NNs use "distinct" bias parameters (e.g., fully-connected layers) and minibatch size is one.
- For NNs with "shared" bias parameters (e.g., convolutional layers) and general minibatch sizes, we prove that AD is always correct if AD uses proper proxy derivatives for activation functions.
- For NNs with non-pointwise activation functions (i.e., maxpools), we provide a sufficient condition for the correctness of AD. We empirically verify that this condition holds in practical scenarios.

Overview

	Distinct bias params	Shared bias params	Minibatch of size one	Choice of proxy derivatives	Without maxpools	Always correct?
Thm1	✓		✓		✓	✓
Lem2			✓		✓	✗
Lem3	✓				✓	✗
Thm4	✓			$D^-\rho(x)$ (or $D^+\rho(x)$)	✓	✓
Thm6		✓		$D^-\rho(x)$ (or $D^+\rho(x)$)	✓	✓
Lem7		✓		$D^-\rho(x)$ (or $D^+\rho(x)$)		✗

- For cases when AD might not be correct (i.e., ✗), we show sufficient conditions when AD is correct (Thm5 & 8).

Problem Setup



- **Condition 1:** Ψ has "distinct bias parameters" and ϕ is identity.
- **Condition 2:** Ψ has "shared bias parameters" and ϕ is maxpool.

AD on Networks with Condition 1

- The neural networks satisfying Condition 1 cover a wide range of practical network architectures including fully-connected neural networks. Likewise, f_l in Condition 1 can represent attention layers, residual connections, normalization layers (e.g., BN and LN), and their compositions. In addition, Condition 1 allows any pointwise and piecewise-analytic activation functions such as ReLU and HardSigmoid.
- Our first theoretical result **(Thm1)** shows that AD *always* computes a correct derivative for a network Ψ with Condition 1 if
  - Ψ has distinct bias parameters and
  - the minibatch size is one.
- If any of the conditions does not hold, then there exist cases where AD does not compute a correct derivative **(Lem2 & 3)**.
- If we suppose that AD uses the proxy derivative of non-differentiable activation ρ at every non-differentiable point x as the same $D^-\rho(x)$ (or $D^+\rho(x)$), then AD is correct **(Thm4)**.
- Some popular activation functions, such as ReLU6, HardTanh, and HardSigmoid, having two non-differentiable points and PyTorch/Tensorflow use $D^{\text{AD}}\rho(x) = 0$ as their proxy gradients for non-differentiable x. Since they use D^- and D^+ simultaneously, these networks do not satisfy the conditions in Theorem 4.
- Thankfully, we can show a sufficient condition for the correctness of AD under *any* minibatch size and the (possible) absence of distinct bias parameters for a network Ψ with Condition 1 **(Thm5)**.

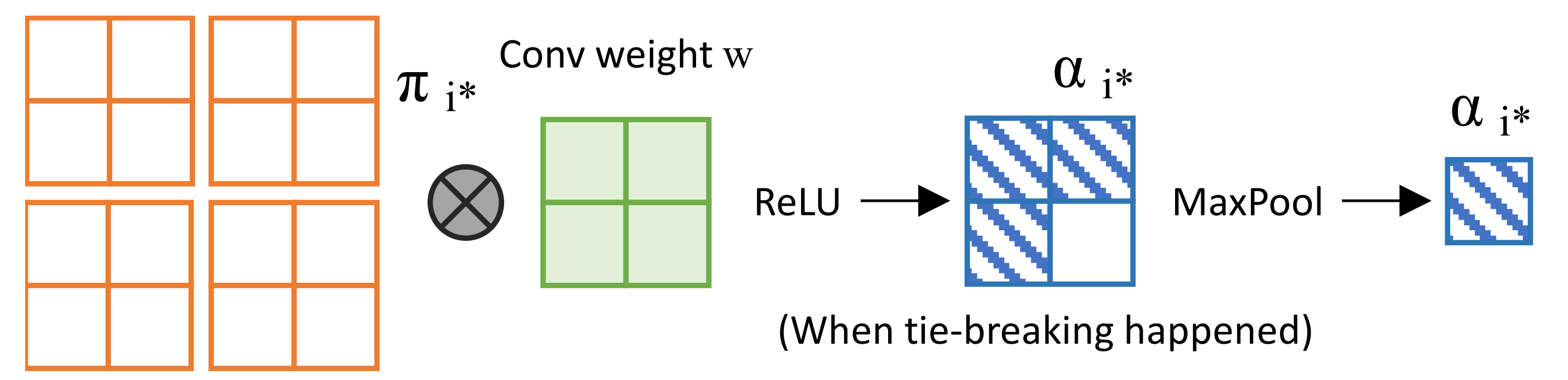
Sufficient condition for fully-connected networks (informal). If the input tensor (hidden dim $\times$ minibatch size) of each layer is linearly independent whenever the layer has any non-differentiable point (e.g., ReLU(0)), then AD is correct. If the input and hidden dimensions are larger than the minibatch size, this condition can be easily satisfied.

AD on Networks with Condition 2

- In practice, we use more complex networks not satisfying Condition 1. For example, a convolutional layer has shared bias parameters. There also exists non-pointwise activations such as maxpools. We extend our results to more complex networks.
- Our theoretical result **(Thm6)** shows that AD *always* computes a correct derivative for a network Ψ satisfying Condition 2 if
  - Ψ has no maxpool (i.e., $\phi(x) = x$) and
  - $D^{\text{AD}}\rho(x) = D^-\rho(x)$ (or $D^+\rho(x)$).

AD on Networks with Condition 2 (Cont.)

- If Ψ has maxpools, then there exist cases where AD does not compute a correct derivative **(Lem7)**.
- However, there exists a sufficient condition where AD computes a correct derivative in this case **(Thm8)**. Furthermore, an efficient algorithm exists to verify the sufficient condition.



Sufficient condition for ConvNets (informal). If there exists x satisfying $\langle \pi_{i^*}, x\rangle > \langle \pi_i, x\rangle$ for all i where $\alpha_i = \alpha_{i^*}$ for every layer where a maxpool has tie-breaking, then AD is correct.

Empricial Verification

- **Scenario 1.** Fully-connected networks with {ReLU6, HardTanh, HardSigmoid} on MNIST:
  - Recap (**Thm4**). These networks do not satisfy the conditions of Thm4; therefore, they do not guarantee the correctness of AD.
  - We can easily verify whether AD is correct by checking the linear independence of the input tensors (**Thm5**).
  - During all training steps, **the networks satisfy the sufficient condition** of Theorem 5. On average, during 9,380 training steps, the non-differentiable points of activation functions were touched 0, 9.8, and 13.8 times for each network.
- **Scenario 2.** ConvNets with maxpools: VGG11, VGG11-BN and ResNet18 on CIFAR-10.
  - Recap (**Lem7**). If a maxpool exists, a network with Condition 2 does not guarantee the correctness of AD.
  - We can verify the correctness of AD by checking the conditions of the events that a tie occurs at maxpool operations (**Thm8**).

Network	Ratio of maxpool tie-breaking	# training steps with the event	Correct?
VGG11	3.2% of 1M max operations	1435.8 steps among 7820 steps	✓
VGG11-BN	4.1% of 1M max operations	3668.5 steps among 7820 steps	✓
ResNet18	4.2% of 2M max operations	3904.8 steps among 7820 steps	✓

- For both scenarios, AD does not guarantee a correct derivative **(Lem2, Lem3 & Lem7)**. However, we empirically verify that AD computes a correct derivative **(Thm5 & Thm8)**.