PyTorch-SSO: Scalable Second-Order Methods

Kazuki Osawa, Tokyo Tech, oosawa.k.ad@m.titech.ac.jp
Yaroslav Bulatov, SPC, yaroslavvb@gmail.com

Codes are available on GitHub
https://github.com/cybertronai/pytorch-sso
https://github.com/cybertronai/autograd-lib

Second-order optimization

First-order optimization (SGD)

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta \nabla L(\theta)$$

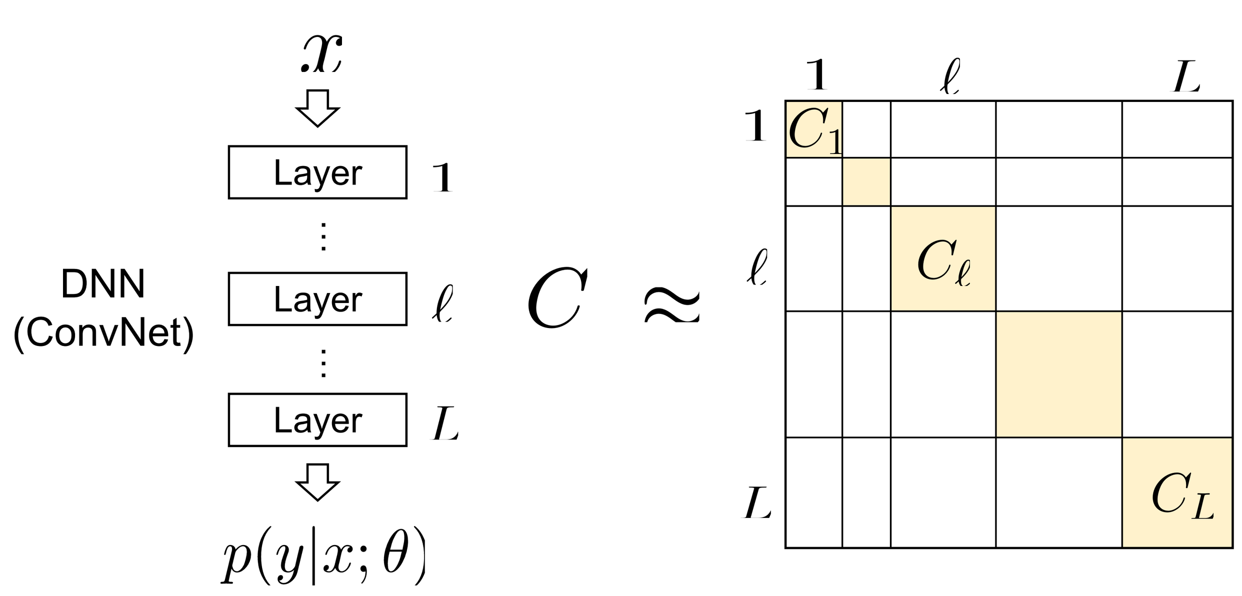
Second-order optimization

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta C^{-1} \nabla L(\theta)$$

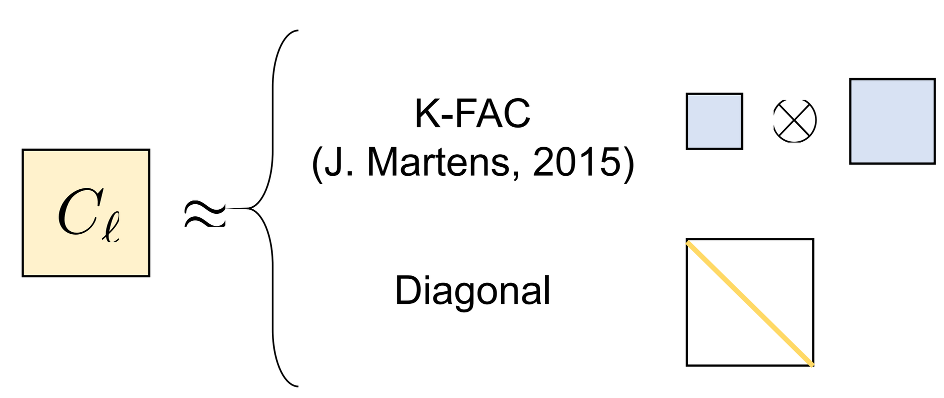
Newton method $C = H = \nabla^2 L(\theta)$
Hessian

Natural Gradient Learning (S. Amari, 1998) $C = F = \mathbb{E}_{x,y}\left[\nabla \log p(y|x;\theta) \nabla \log p(y|x;\theta)^{\mathrm{T}}\right]$
Fisher information matrix

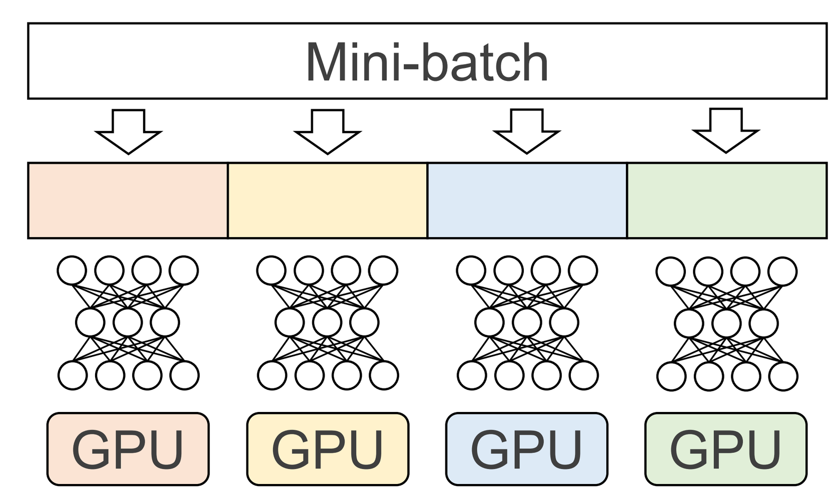
Scalable second-order optimization



Layer-wise block-diagonal approximation



Practical approximation



Distributed training

Using PyTorch autograd

$w_l \leftarrow w_l + g_l$	SGD	$\ldots$	
$w_l \leftarrow w_l + g_l / h_l$	Diagonal Newton	$\{h_1, \ldots, h_L\}$	Hessian diagonals
$w_l \leftarrow w_l + H_l^{-1} g_l$	Newton	$\{H_1, \ldots, H_L\}$	Layer Hessians
$w_l \leftarrow w_l + C_l^{-1} g_l$	Natural Gradient	$\{C_1, \ldots, C_L\}$	Layer Fisher matrices

Bonus: single-pass estimation of

1. OpenAI's gradient noise $= \frac{\mathrm{trace}(H\Sigma)}{gHg'}$
2. Berkeley's gradient diversity $= \frac{E|g^2|}{|E[g]|^2}$
3. per-example gradients

Problem: autograd

1. Needs O(L) backward calls
2. Doesn't use structure

Solution: **autograd_lib**

1. Uses O(1) backward calls
2. Uses structure

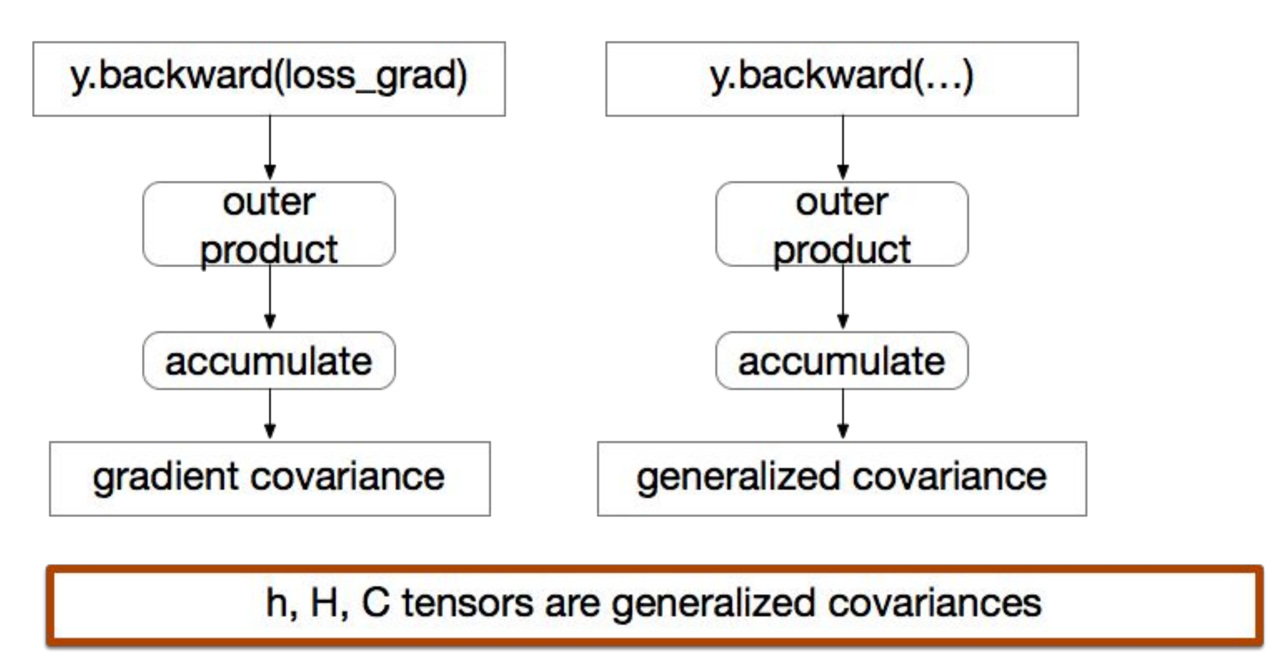
Example of rank-2 structure

$f(x) = \mathrm{ReLU}$ network
$\mathrm{loss}(x) = f^T(x) f(x)$
$H_L(x) = A_l(x) \otimes B_l(x)$ $\quad 2d^2 \quad 8$ KB (for d=1000)
expanded $H_l(x)$ $\quad d^4 \quad 4$ TB

Structures

d=input size (Linear)
d=Kw*Kh*Ci (Conv2d)

structure rank	example	storage cost	preconditioning cost
1	batch-norm	d	d
2	diagonal Newton	d^2	d^2
2	KFAC	$2d^2$	$O(d^3)$
3	Isserlis	$3d^2$	
4	naive	d^4	$O(d^6)$



Papers

1. **Distributed K-FAC with an extremely large mini-batch size of 131K on ImageNet** (By Chainer version of this library which scales to 1024 GPUs). Kazuki Osawa et al, "Large-Scale Distributed Second-Order Optimization Using Kronecker-Factored Approximate Curvature for Deep Convolutional Neural Networks", **IEEE/CVF CVPR 2019**.
2. **Distributed natural gradient learning for Bayesian deep learning on ImageNet** (By this library). Kazuki Osawa et al, "Practical Deep Learning with Bayesian Principles", **NeurIPS 2019**.