北京大学 PEKING UNIVERSITY

Stochastic Gradient Descent

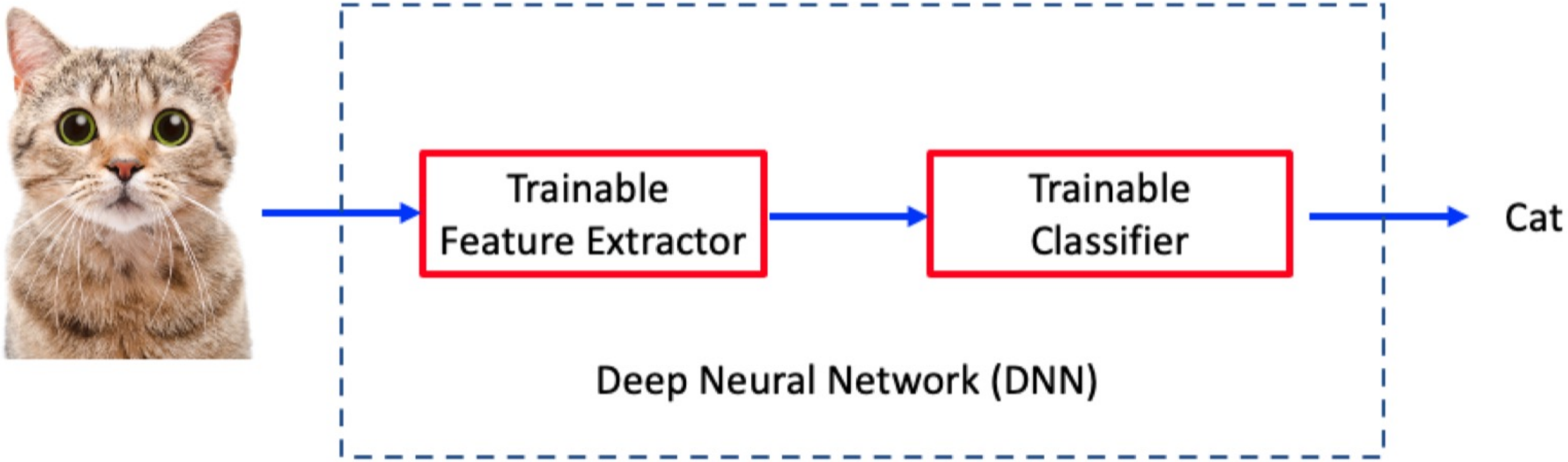
Kun Yuan

Center for Machine Learning Research @ Peking University

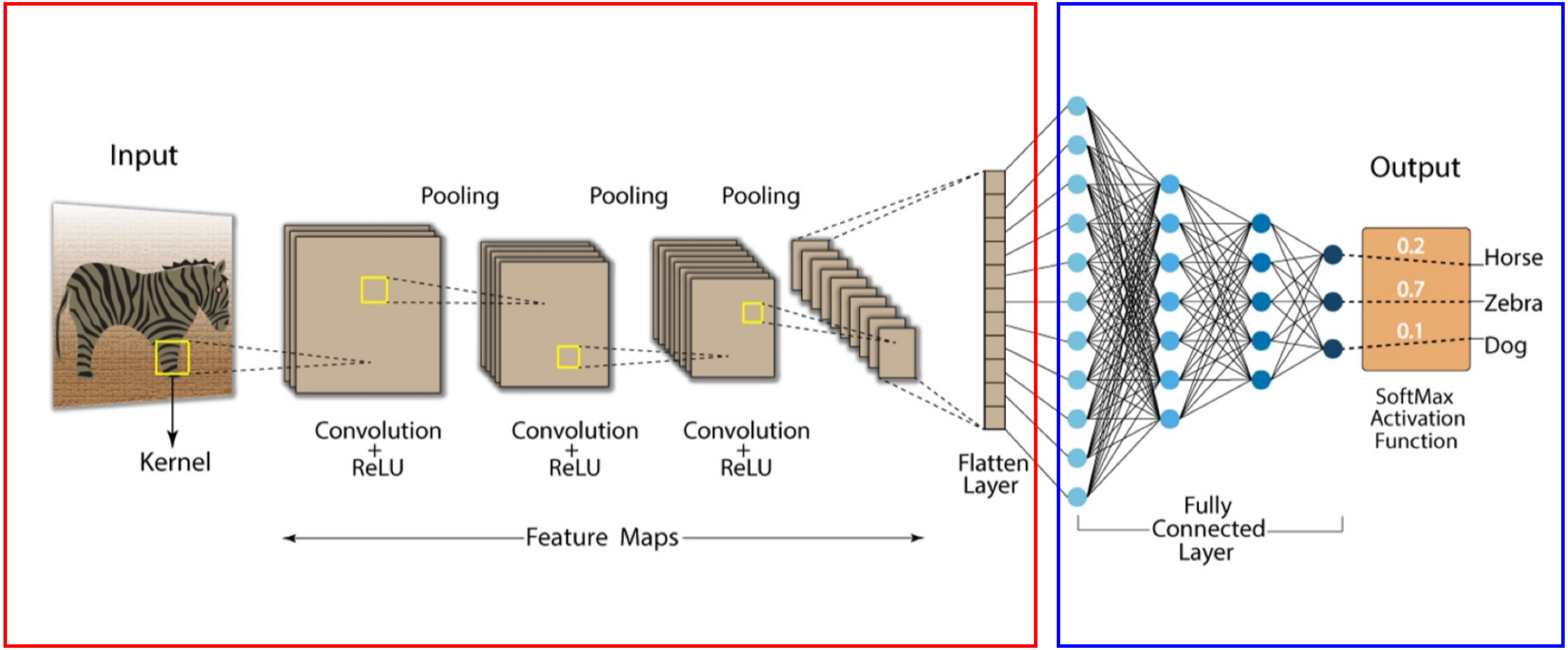
Deep neural network (DNN)

- A deep neural network (DNN) typically includes a **feature extractor** and a **classifier**

- Well-trained DNN can make precise predictions



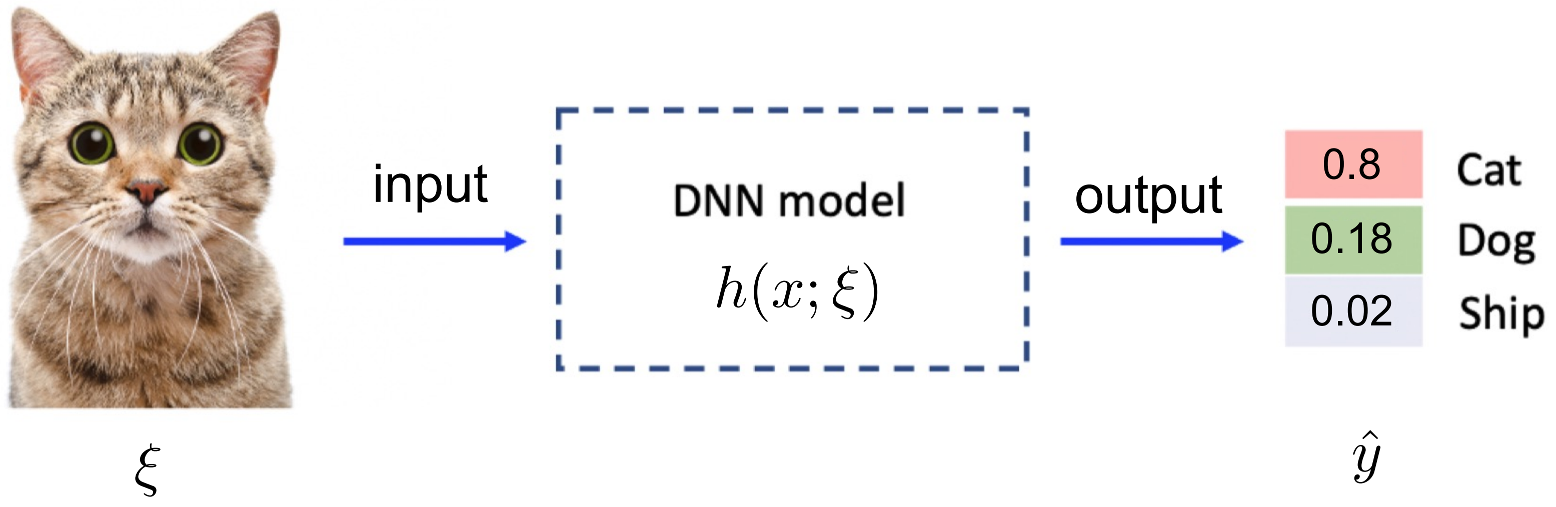
Example: convolutional neural network



Feature extractor **Classifier**

DNN model

- We model DNN as $h(x;\xi): \mathbb{R}^d \to \mathbb{R}^c$ that maps input data ξ to a probability $\hat{y}$
  - $x \in \mathbb{R}^d$ is the DNN model parameter to be trained
  - ξ is a random input data sample
  - c is the number of classes



Training DNN can be formulated into an optimization problem

- Define $L(\hat{y}, y) = -\sum_{j=1}^{d} y_{[j]} \log(\hat{y}_{[j]})$ as the loss function to measure the difference between predictions and the ground-truth label, where $y_{[j]}$ is the j-th element in y

- The model parameter $x^\star$ can be achieved by solving the following optimization problem

$$x^\star = \arg\min_{x\in\mathbb{R}^d} \left\{ \mathbb{E}_{\underbrace{(\xi,y)\sim\mathcal{D}}_{\text{data distribution}}} \left[L\Big(\underbrace{h(x;\xi)}_{\text{prediction}}, \underbrace{y}_{\text{real label}}\Big)\right]\right\}$$

Training DNN can be formulated into an optimization problem

- The model parameter $x^\star$ can be achieved by solving the following optimization problem

$$x^\star = \arg\min_{x\in\mathbb{R}^d} \Big\{ \mathbb{E}_{\underbrace{(\xi,y)\sim\mathcal{D}}_{\text{data distribution}}} \Big[L\big(\underbrace{h(x;\xi)}_{\text{prediction}}, \underbrace{y}_{\text{real label}}\big) \Big] \Big\}$$

- If we define $\boldsymbol{\xi} = (\xi, y)$ and $F(x;\boldsymbol{\xi}) = L(h(x;\xi), y)$, the above problem becomes

$$x^\star = \arg\min_{x\in\mathbb{R}^d} \{\mathbb{E}_{\boldsymbol{\xi}\sim\mathcal{D}}[F(x;\boldsymbol{\xi})]\}$$

- Most optimization researchers use the second formulation as the staring point to develop algorithms

DNN model is notoriously difficult to train

- Highly-nonconvex cost functions; cannot find global minima; trapped into local minimum

- The model size is large, i.e., $x \in \mathbb{R}^d$ is of extremely high dimensions

- The size of the dataset is huge

DNN training = Non-convex training + Huge dimensions + Huge dataset

- **Efficient** and **scalable** distributed learning approaches are in urgent need

Stochastic optimization

- Consider the stochastic optimization problem:

$$\min_{x \in \mathbb{R}^d} \quad f(x) = \mathbb{E}_{\xi \sim \mathcal{D}}[F(x;\xi)]$$

  - ξ is a random variable indicating data samples
  - $\mathcal{D}$ is the data distribution; unknown in advance
  - $F(x;\xi)$ is differentiable in terms of x

- Many applications in signal processing and machine learning

Example: deep neural network

- Recall the DNN training problem

$$\min_{x \in \mathbb{R}^d} \quad \frac{1}{m} \sum_{i=1}^{m} L(h(x; a_i), b_i)$$

  which is a finite-sum problem

- Suppose we have infinite data (a, b) following distribution D, the above problem becomes

$$\min_{x \in \mathbb{R}^d} \quad f(x) = \mathbb{E}_{(a,b)\sim\mathcal{D}} L(h(x; a), b)$$

  where data pair (a, b) can be regarded as sample ξ.

Stochastic gradient descent

- Recall the problem

$$\min_{x \in \mathbb{R}^d} \quad f(x) = \mathbb{E}_{\xi \sim \mathcal{D}}[F(x;\xi)]$$

- Closed-form of $f(x)$ is unknown; gradient descent is not applicable

- Stochastic gradient descent (SGD):

$$x_{k+1} = x_k - \gamma \nabla F(x_k; \xi_k), \quad \forall k = 0, 1, \cdots$$

  where ξ_k is a data realization sampled at iteration k.

- Since $\{x_k\}$ are random, all iterates $\{x_k\}$ are also random

Assumption

Let $\mathcal{F}_k = \{x_k, \xi_{k-1}, x_{k-1}, \cdots, \xi_0\}$ be the filtration containing all historical variables at and before iteration k (except for ξ_k).

Assumption 1

Given the filtration $\mathcal{F}_k$, we assume

$$\mathbb{E}[\nabla F(x_k; \xi_k) | \mathcal{F}_k] = \nabla f(x_k)$$
$$\mathbb{E}[\|\nabla F(x_k; \xi_k) - \nabla f(x_k)\|^2 | \mathcal{F}_k] \leq \sigma^2$$

Implying **unbiased** stochastic gradient and **bounded** variance.

Convergence: smooth and non-convex scenario

Theorem 1

Suppose $f(x)$ is L-smooth and Assumption 1 holds. If $\gamma \le 1/L$, SGD will converge at the following rate

$$\frac{1}{K+1}\sum_{k=0}^{K}\mathbb{E}[\|\nabla f(x_k)\|^2] \le \frac{2\Delta_0}{\gamma(K+1)} + \gamma L\sigma^2,$$

where $\Delta_0 = f(x_0) - f^\star$.

- SGD **cannot** converge to stationary point with constant learning rate
- Smaller learning rate γ or variance σ^2 leads to smaller convergence error

Image Classification

Cifar-10 dataset

50K training images

10K test images

DNN model: ResNet-18

GPU: Tesla V100

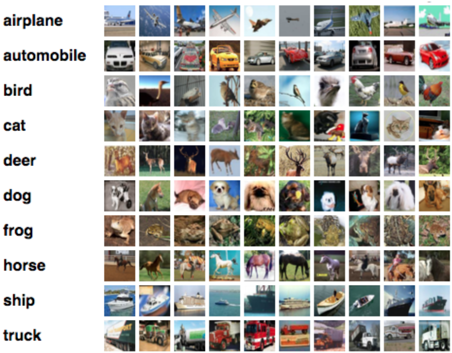
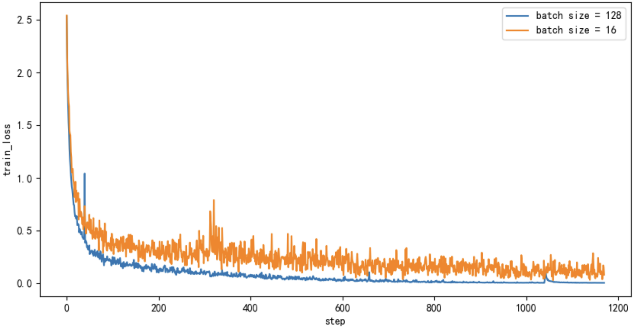


Image Classification

Large batch-size helps training.



Convergence: smooth and non-convex scenario

Corollary 1

Suppose $f(x)$ is L-smooth and Assumption 1 holds. If γ is chosen as

$$\gamma = \left[\left(\frac{2\Delta_0}{(K+1)L\sigma^2} \right)^{-\frac{1}{2}} + L \right]^{-1},$$

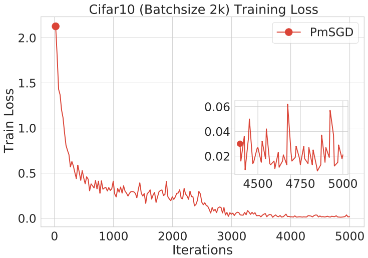
SGD will converge at the following rate

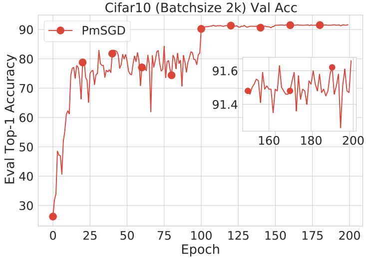
$$\frac{1}{K+1} \sum_{k=0}^{K} \mathbb{E}[\|\nabla f(x_k)\|^2] \le \sqrt{\frac{8L\Delta_0\sigma^2}{K+1}} + \frac{2L\Delta_0}{K+1}.$$

where $\Delta_0 = f(x_0) - f^\star$.

- Decaying rate leads to exact convergence to stationary point
- When $\sigma^2 = 0$, the above rate **reduces to GD**; rate is tight!
- $O(\sqrt{\sigma^2/K})$ is the dominant rate

Image Classification

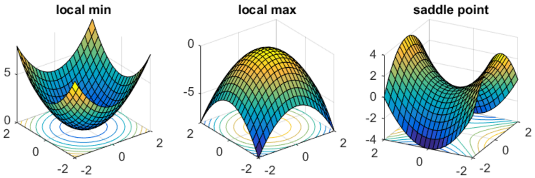




Convergence: smooth and non-convex scenario

$$\frac{1}{K+1}\sum_{k=0}^{K}\mathbb{E}\|\nabla f(x_k)\|^2 = O\left(\sqrt{\frac{L\sigma^2}{K+1}} + \frac{L}{K+1}\right)$$

- When iteration $K \to \infty$, it holds that $\mathbb{E}\|\nabla f(x_K)\|^2 \to 0$
- $\mathbb{E}\|\nabla f(x_K)\|^2 \to 0$ implies SGD converges to a stationary solution
- A stationary solution can be local min, local max, or saddle point[4]



[4] Image source: from Prof. Rong Ge's online post

Convergence: smooth and non-convex scenario

- Generally speaking, approaching the stationary solution is the best result we can get for SGD; no guarantee to approach the global minimum

- Empirically, SGD performs extremely well when training DNN

- Recent advanced studies show SGD can escape local maximum, saddle point, and even "sharp" local minimum, see, e.g., (Ge et al., 2015; Sun et al., 2015; Jin et al., 2017; Du et al., 2018, 2019; Kleinberg et al., 2018)

- SGD can even find global minimum under certain conditions, e.g. the PL condition (Karimi et al., 2016)

  However, we will skip these interesting results in this lecture

References I

R. Ge, F. Huang, C. Jin, and Y. Yuan, "Escaping from saddle points—online stochastic gradient for tensor decomposition," in *Conference on learning theory*. PMLR, 2015, pp. 797–842.

J. Sun, Q. Qu, and J. Wright, "When are nonconvex problems not scary?" *arXiv preprint arXiv:1510.06096*, 2015.

C. Jin, R. Ge, P. Netrapalli, S. M. Kakade, and M. I. Jordan, "How to escape saddle points efficiently," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1724–1732.

S. S. Du, X. Zhai, B. Poczos, and A. Singh, "Gradient descent provably optimizes over-parameterized neural networks," *arXiv preprint arXiv:1810.02054*, 2018.

S. Du, J. Lee, H. Li, L. Wang, and X. Zhai, "Gradient descent finds global minima of deep neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 1675–1685.

References II

B. Kleinberg, Y. Li, and Y. Yuan, "An alternative view: When does sgd escape local minima?" in *International Conference on Machine Learning*. PMLR, 2018, pp. 2698–2707.

H. Karimi, J. Nutini, and M. Schmidt, "Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2016, pp. 795–811.