Video 34

Sunday, November 29, 2020 10:30 PM

ResNet

feedforward

Goal: train DNNs (convolutional neural networks) w/ many layers.

trained on ImageNet:
- GoogleNet architecture → 20 layers (excluding pooling)
- ResNet architecture → 150 layers

ResNet: deep residual network.

Intuition: Jacobian of a RNN

$$\frac{\partial \vec{h}^{(T)}}{\partial \vec{h}^{(1)}} = \frac{\partial \vec{h}^{(T)}}{\partial \vec{h}^{(T-1)}} \cdots \frac{\partial \vec{h}^{(2)}}{\partial \vec{h}^{(1)}}$$

product of the Jacobian of individual layers.

Multiplying all these individual matrices is why gradients explode / vanish unless we keep each matrix close to the identity.

⇒ This formula applies to feed-forward NNs (conv neural nets)

Feed forward neural network:

$$\vec{h} = \phi(\vec{z})$$

↑ activation function

$$\vec{z} = W\vec{x} + \vec{b}$$

Use this as an architectural primitive called the residual block.

Residual block:

$$\vec{y} = \vec{x} + \mathcal{F}(\vec{x})$$

$\mathcal{F}$: residual function

Special case:

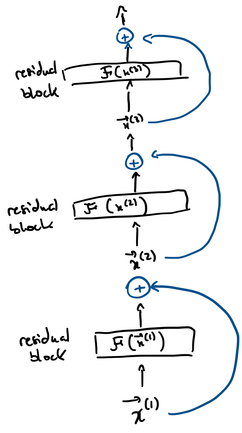
$$\mathcal{F}(\vec{x}) = \phi(\vec{z}) = W\vec{x} + b$$

Typically: residual function is a small / shallow neural network w/ 1–3 hidden layers.

ResNet:

Deep residual network

Each residual block has separate trainable parameters



Why does this architecture helps prevent gradients exploding / vanishing?

↳ Jacobian of a residual block should be close to the identity.

Backpropagation rules:

$$\frac{\partial \ell}{\partial \vec{x}^{(l)}} = \frac{\partial \ell}{\partial \vec{x}^{(l+1)}} \frac{\partial \mathcal{F}}{\partial \vec{x}} + \frac{\partial \ell}{\partial \vec{x}^{(l+1)}}$$

$$= \frac{\partial \ell}{\partial \vec{x}^{(l+1)}} \left(\frac{\partial \mathcal{F}}{\partial \vec{x}} + I \right)$$

I: identity matrix

if $\frac{\partial \mathcal{F}}{\partial \vec{x}} \to 0$, the sum is close to the identity

⇒ error signal / gradients will not explode or vanish.