Advanced Machine Learning

Likhit Nayak

Steepest Descent

In gradient descent, we fix the learning rate in advance:

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta)$$

In steepest descent, we find the learning rate that minimizes the following function:

$$J(\theta - \eta \cdot \nabla_\theta J(\theta))$$

Newton’s Method

Using second-order derivatives to minimize the objective function:

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x},\mathrm{y}\sim\hat{p}_{\mathrm{data}}(\boldsymbol{x},y)}[L(f(\boldsymbol{x};\boldsymbol{\theta}),y)] = \frac{1}{m}\sum_{i=1}^{m} L(f(\boldsymbol{x}^{(i)};\boldsymbol{\theta}),y^{(i)})$$

We use second-order Taylor series expansion :

$$J(\boldsymbol{\theta}) \approx J(\boldsymbol{\theta}_0) + (\boldsymbol{\theta}-\boldsymbol{\theta}_0)^\top \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0) + \frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\theta}_0)^\top \boldsymbol{H}(\boldsymbol{\theta}-\boldsymbol{\theta}_0)$$

BFGS Algorithm

Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm attempts to address the computational burdens of Newton's algorithm. The Newton's update is given by:

$$\boldsymbol{\theta}^* = \boldsymbol{\theta}_0 - \boldsymbol{H}^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_0)$$

Instead of calculating ***H*** and inverting it, we just approximate it using a positive-definite matrix ***M***:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \epsilon^* \boldsymbol{M}_t \boldsymbol{g}_t$$

Batch Normalization

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

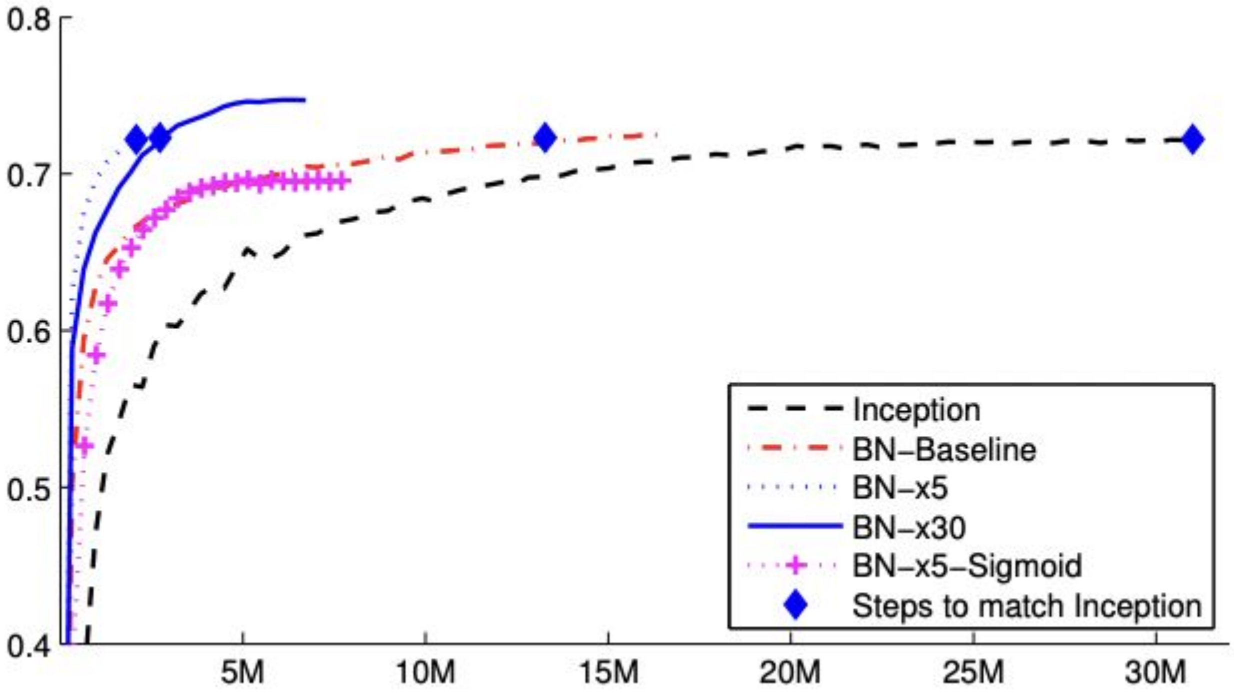
$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

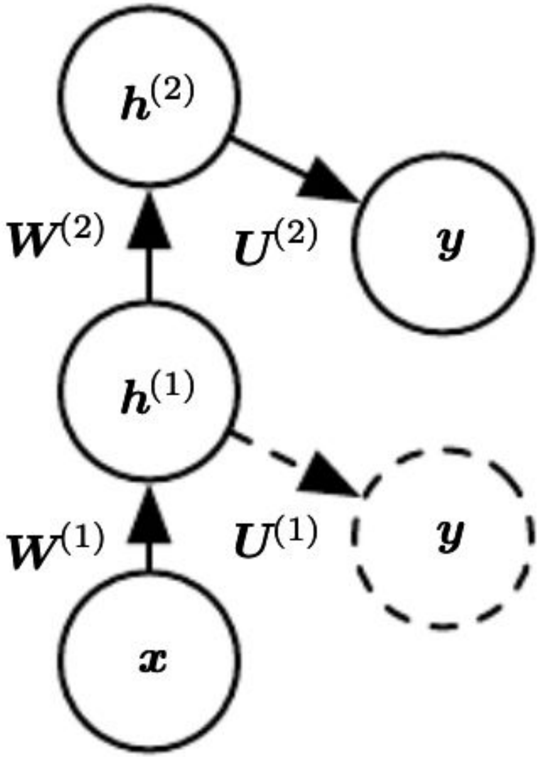
Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *International conference on machine learning*. pmlr, 2015.

Batch Normalization

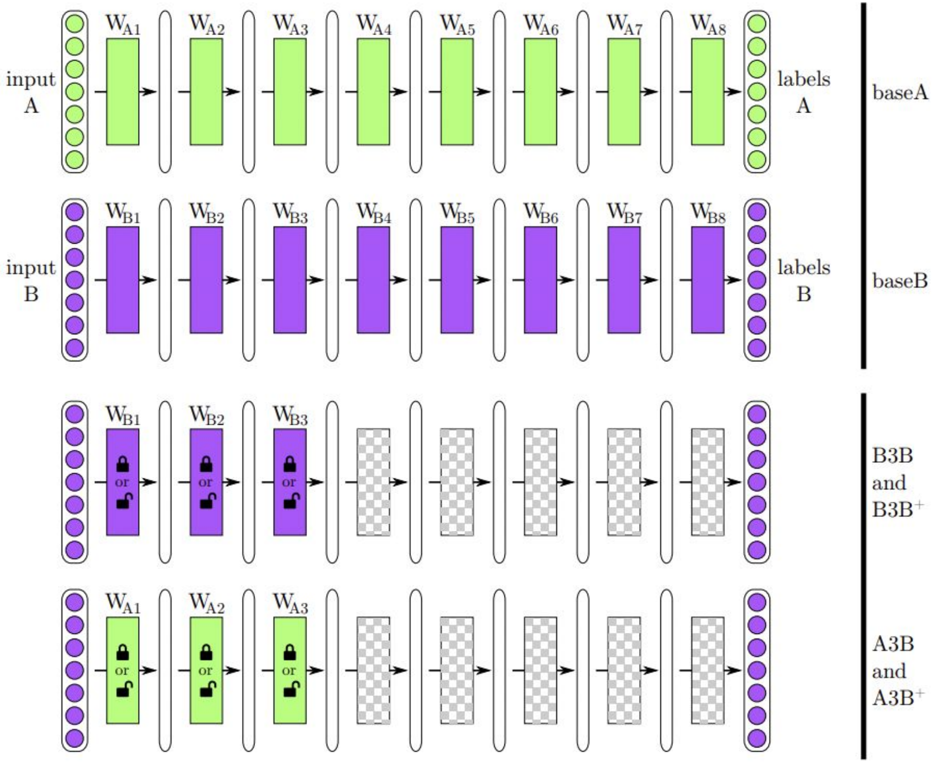


Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *International conference on machine learning*. pmlr, 2015.

Greedy supervised pre-training

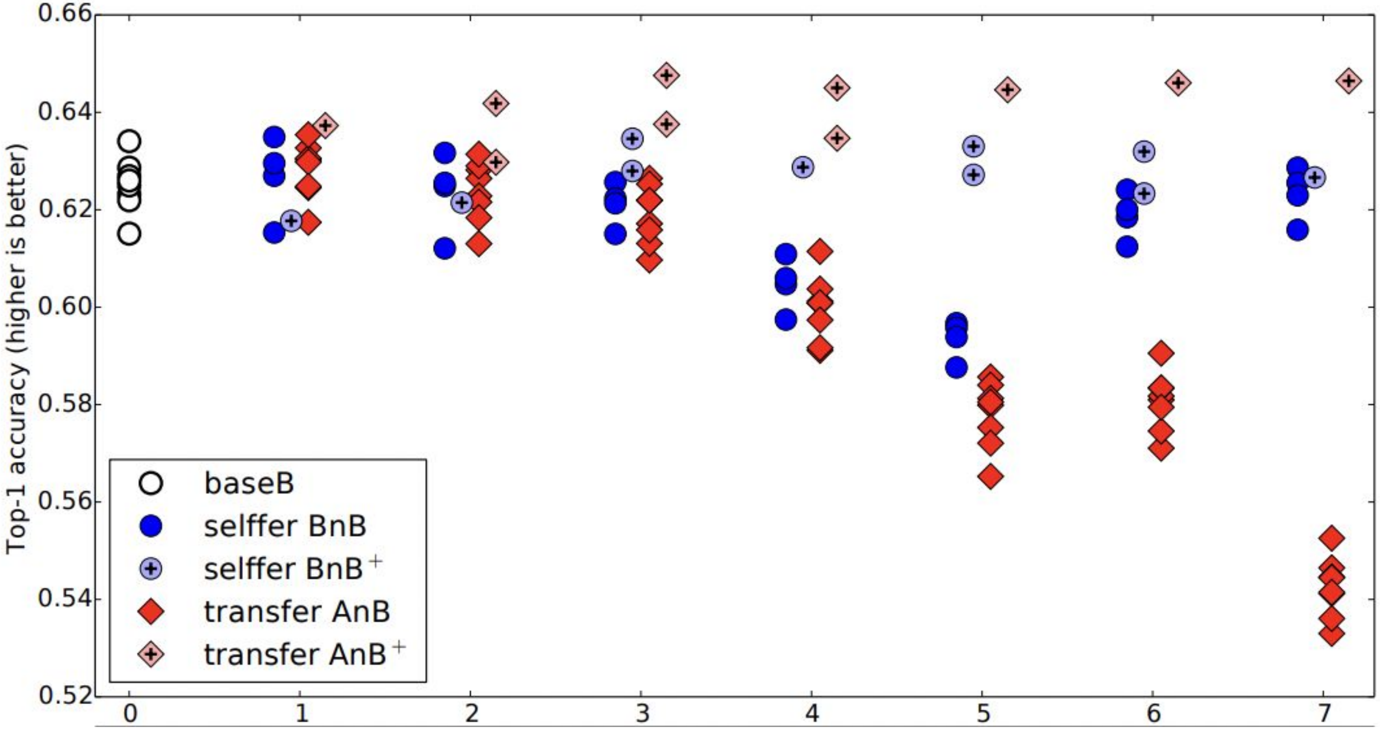


Transfer Learning



Yosinski, Jason, et al. "How transferable are features in deep neural networks?." *Advances in neural information processing systems* 27 (2014).

Transfer Learning



Yosinski, Jason, et al. "How transferable are features in deep neural networks?." *Advances in neural information processing systems* 27 (2014).