Deep Networks

Overfitting + Generalization

p parameters, n datapoints, d dimensions

$p \approx d$. ← Classic

$p \gg nd$ ← Modern NNs

$p \simeq 1$ trillion $\qquad n = 60000,\ d = 28 \times 28$

Avoiding Overfitting

- Bagging
- Regularization

$$\hookrightarrow L(x, y, \theta) + \frac{\lambda}{2}\|\theta\|_2^2$$

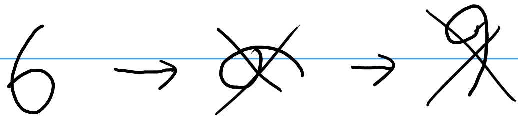
$$\xrightarrow{\nabla_\theta} \nabla_\theta L(x, y, \theta_{t-1}) + \lambda \theta_{t-1}$$

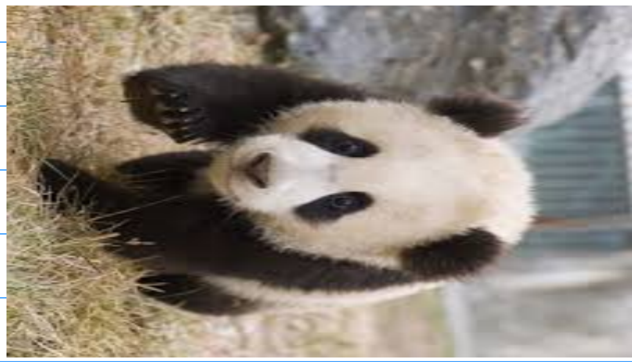
$$\theta_t = \theta_{t-1} - \eta\left(\nabla_\theta L(x, y, \theta_{t-1}) + \lambda \theta_{t-1}\right)$$

$$\theta_t = (1 - \eta\lambda)\theta_{t-1} - \eta \nabla_\theta L(x, y, \theta_{t-1})$$

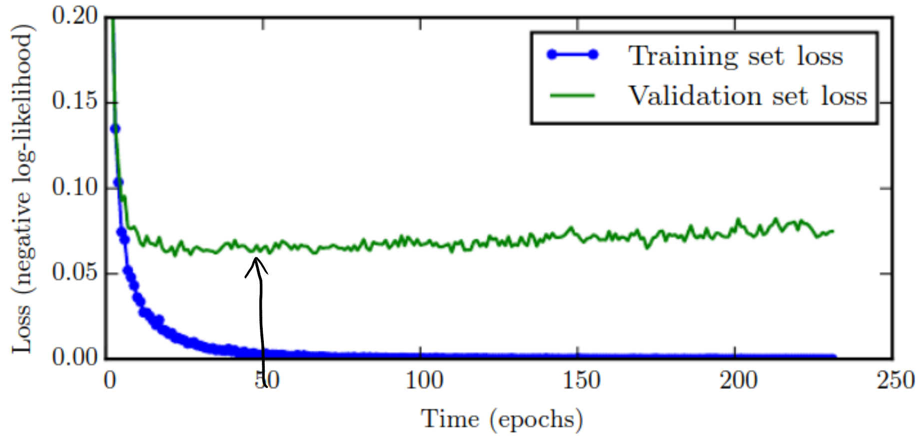
↑ Weight decay

- Data Augmentation





Early Stopping

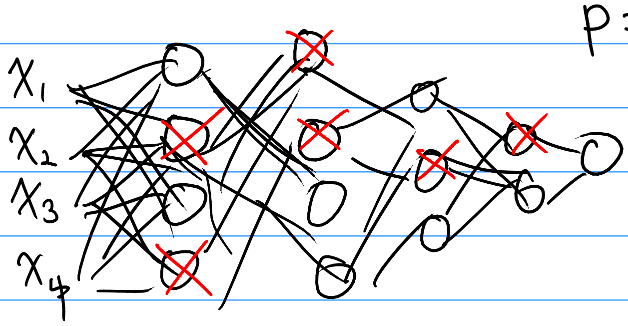


Checkpointing

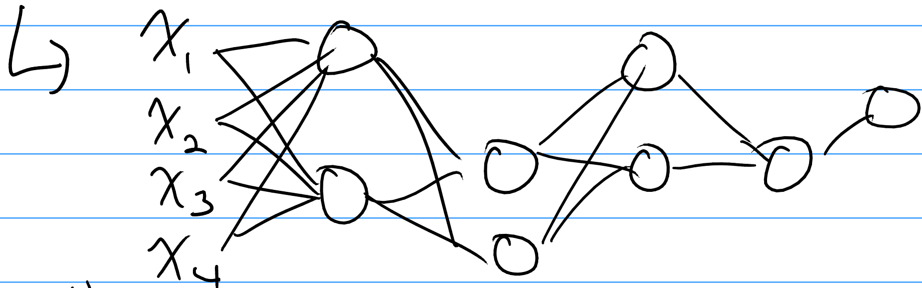
Dropout

Keep each node w.p. $p > 0$, when training each point

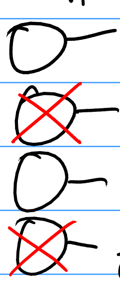
$p = 0.5$



↳



$h^{(1)}$



$\frac{1}{0.5} 1 \cdot h_1^{(1)}$

$\frac{1}{0.5} 0 \cdot h_2^{(1)}$

$\frac{1}{0.5} 1 \cdot h_3^{(1)}$

$\frac{1}{0.5} 0 \cdot h_4^{(1)}$

$$z^{(2)} = \underbrace{W^{(2)} h^{(1)}}_{\approx p \text{ times its prev value (before Dropout)}} + b^{(2)}$$

Multipy $h^{(1)}$ by $\frac{1}{p}$. (Inverted dropout)

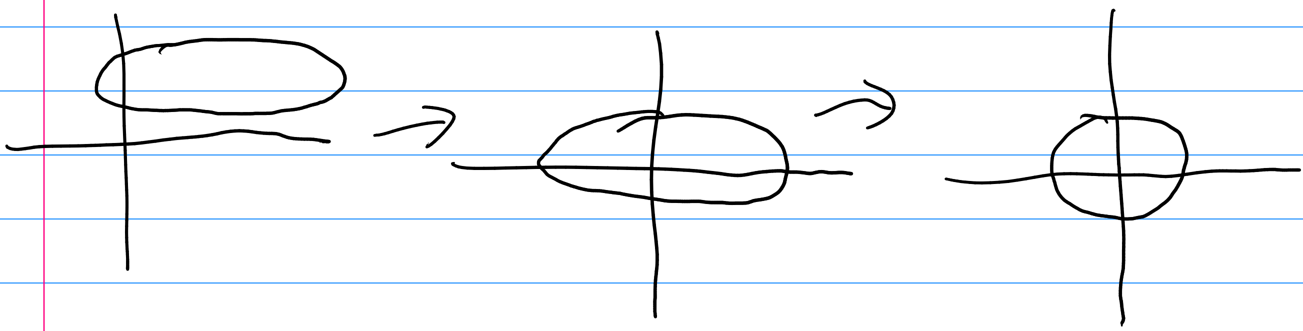
Test time: No dropout, no scaling

Normalization

Normalize features before training

$\mu = \frac{1}{n}\sum X_i$. $X_i = X_i - \mu$. $\sigma_j^2 = \frac{1}{n}\sum X_{ij}^2$. $X_{ij} = X_{ij}/\sigma_j$

$\forall i$ $\forall i,j$



Batch norm

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

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

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

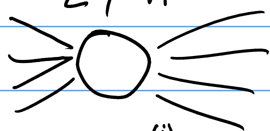
$$\sigma_\mathcal{B}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_\mathcal{B})^2 \qquad \text{// mini-batch variance}$$

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

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

learnable params

z | h



$z^{(i)} = W h^{(i-1)} + b^{(i)}$

$h^{(i)} = f(z^{(i)})$

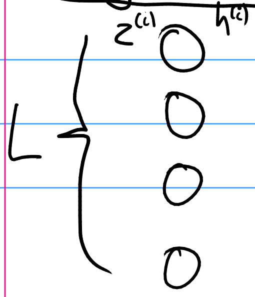
Minibatch SGD

- Process 64 pts at once. (128, 256)

Applied to each neuron individually

Averaging over a minibatch.

Layer norm



For each datapoint:

$\mu^{(i)} = \frac{1}{L}\sum_{j=1}^{L} z_j^{(i)}$ $\quad z_j^{(i)} = z_j^{(i)} - \mu^{(i)}$

$\sigma^{(i)2} = \frac{1}{L}\sum z_j^{(i)2}$ $\quad z_j^{(i)} = \frac{z_j^{(i)}}{\sigma^{(i)2}}$

Averaged over a layer

Optimization

First-order methods: 1st derivative
Second-order " " 2nd " "

Gradient Descent "Batch"

$$\theta \leftarrow \theta - \eta \cdot \underbrace{\frac{1}{n}\sum_{i=1}^{n} \nabla_\theta l(x_i, y_i, \theta)}$$

$$\mathbb{E}_{(x,y)\sim D}\left[\nabla_\theta l(x,y,\theta)\right]$$

η ← hyper parameter

Stochastic GD

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta l(x_i, y_i, \theta)$$

Shuffling after each epoch

Mini-batch SGD

$$\theta \leftarrow \theta - \eta \cdot \frac{1}{|B|} \cdot \sum_{i \in B} \nabla_\theta l(x_i, y_i, \theta)$$

$\underbrace{x_1, x_2,} \underbrace{x_3, \ldots,} \underbrace{x_n}$

$|B| \approx 64$ to 256

Challenges

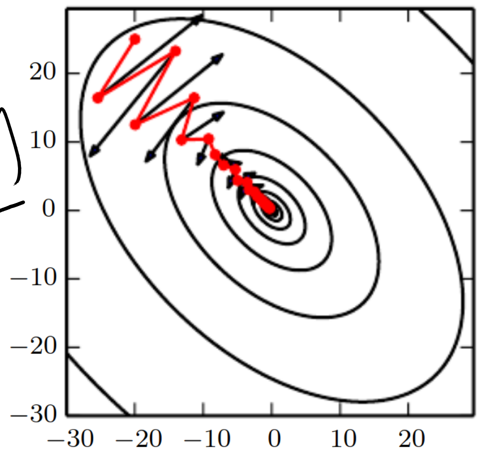
- η?
- LR schedules ← doesn't "adapt" to data
- Can we have diff η for diff coords?

"Momentum" $\gamma < 1$.

$\gamma = 0.9?$

$$V_t = \gamma V_{t-1} + (1-\gamma)\eta \underbrace{\frac{1}{|B|}\sum \nabla \ell(x_i, y_i, \theta)}_{g_t}$$

$$\theta \leftarrow \theta - V_t$$



$$V_t = 0.1 g_t + 0.1 \cdot 0.9 g_{t-1} + 0.1 \cdot 0.9^2 g_{t-2} + \cdots + 0.1 \cdot 0.9^{t-1} g_1$$

Black: Gradient
Red: momentum + SGD

Total coeff: $1-\gamma^t \simeq 1$ when t large

Nesterov Momentum/Accelerated Gradient

$$V_t = \gamma V_{t-1} + (1-\gamma)\eta \frac{1}{|B|}\sum \nabla \ell(x_i, y_i, \theta - \gamma V_{t-1})$$
$$\theta = \theta - V_t$$

Adaptive Learning Rates

Change LR over algo.
Based on "importance" of each param.

- Lot of updates $\Rightarrow$ low LR for param
- Few updates $\Rightarrow$ large LR for param

$g_t \in \mathbb{R}^p$ is gradient at time t.

SGD $\rightarrow$ $\theta_{t+1,i} = \theta_{t,i} - \eta g_{t,i}$ $\leftarrow$ coord i update

Define $G_{t,i} = \sum_{j=1}^{t} g_{j,i}^2$ (sum of squared grad updates)

AdaGrad

$$\rightarrow \theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,i}+\varepsilon}} g_{t,i}$$

$\varepsilon \approx 10^{-8}$

RMSProp: $G_{t,i} = 0.9 G_{t-1,i} + 0.1 g_{t,i}^2$

Momentum, but on 2nd moment (grad^2) rather than 1st moment

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,i}+\varepsilon}} g_{t,i}$$

Adam

- Momentum + RMSProp (plus bias correction)
- β_1, β_2 hyper params

$m_{t,i} = \beta_1 m_{t-1,i} + (1-\beta_1) g_{t,i}$ ← Momentum

$V_{t,i} = \beta_2 V_{t-1,i} + (1-\beta_2) g_{t,i}^2$ ← RMS Prop updates

$\hat{m}_{t,i} = \frac{m_{t,i}}{1-\beta_1^t}$, $\hat{V}_{t,i} = \frac{V_{t,i}}{1-\beta_2^t}$ ← Bias correction

$$\theta_{t+1,i} = \theta_t - \frac{\eta}{\sqrt{\hat{V}_{t,i}}+\varepsilon} \hat{m}_{t,i}$$

$\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$