CSC413 HW2

1. Optimization

$$L = \frac{1}{n}\|X\hat{w} - t\|_2^2, \quad X \in \mathbb{R}^{n\times d}, \quad \hat{w} \in \mathbb{R}^{d\times 1}, \quad t \in \mathbb{R}^{n\times 1}$$

1.1. Stochastic Gradient Descent (SGD)

$$w_{t+1} \leftarrow w_t - \eta \nabla_{w_t} L_i(x_i, w_t), \quad 1 \le i \le n.$$

1.1.1 Minimum Norm Solution

result from HW1, Q3.4: $\underset{d\times 1}{\frac{\partial L}{\partial \hat{w}}} = \bar{\hat{w}} = \frac{\partial}{\partial \hat{w}}\left(\frac{1}{n}\|X\hat{w} - t\|_2^2\right)$

$= \frac{2}{n}\underset{d\times n}{X^T}\underset{n\times 1}{(X\hat{w} - t)}$, in the SGD case, $\underset{d\times 1}{\bar{\hat{w}}_t} = 2\underset{d\times 1}{x_i}(\underset{1\times d}{x_i^T}\underset{d\times 1}{\hat{w}_t} - \underset{1\times 1}{t_i})$

Starting from zero initialization,

$\bar{\hat{w}}_t(0) = -2\underset{d\times 1}{x_i}\underset{1\times 1}{t_i}$, <u>gradient is in the direction of x_i.</u>

Being a row in $X \in \mathbb{R}^{n\times d}$, x_i^T is in the row space of X, so x_i is in the column space/span of X.

therefore, <u>gradient is in the span of X.</u>

Update steps of SGD never leaves the span of X, since for $\bar{\hat{w}}_t$ always in direction of x_i, no matter what $(x_i^T\hat{w}_t - t_i)$ is.

with the assumption that $X\hat{w} = t$ eventually,

$w_{t+1} \leftarrow \underbrace{w_t}_{\in \text{span} X} - \eta \underbrace{\bar{\hat{w}}_{t,i}}_{\in \text{span} X}$, so $w_{t+1} \in \text{span } X$, let $C^{n\times 1} \in \mathbb{R}$ denote the final coefficients,

$$\underset{n\times d}{X}(\underset{d\times n}{X^T}\underset{n\times 1}{C}) = \underset{n\times 1}{t}$$

$$C = (XX^T)^{-1}t$$

$$w^{**} = X^T(XX^T)^{-1}t = w^*, \text{ same as before!}$$

1.1.2 Mini-batch SGD

$B \in \mathbb{R}^{b\times d}$, the loss at each timestep is: $L(B_i, w_t)$,

the question boils down to whether the update $\nabla_{w_t} L(B, w_t) \in \text{span } X$

$\underset{d\times 1}{\frac{\partial L}{\partial \hat{w}_t}} = \bar{\hat{w}}_t = \frac{2}{b}\underset{d\times b}{B_i^T}(\underset{b\times d}{B_i}\underset{d\times 1}{W_t} - \underset{b\times 1}{t_i})$,

$\bar{\hat{w}}(0) = -\frac{2}{b}\underset{d\times b}{B_i^T}\underset{b\times 1}{t_i}$, derivative in the span of B_i,

$B_i^T t_i$ is a linear combination of columns in X, so still in span X,

using the same reasoning as before, yes.

1.2 Adaptive Methods

$$G_{i,t} = G_{i,t-1} + (\nabla_{w_{i,t}} L(w_{i,t}))^2$$

$$w_{i,t+1} = w_{i,t} - \frac{\eta}{\sqrt{G_{i,t}+\varepsilon}}\nabla_{w_{i,t}} L(w_{i,t})$$

1.2.1 Minimum Norm Solution

Similar as before, $\bar{\hat{w}} = 2\begin{bmatrix}2\\1\end{bmatrix}(0-2) = \begin{bmatrix}-8\\-4\end{bmatrix}$ at zero initialization,

update is <u>different</u> for the two weights,

for first weight: $w_{1,1} = \cancel{w_{1,0}}^{0} - \frac{\eta}{8+\varepsilon}(-8)$

for second weight: $w_{2,1} = \underset{0}{\cancel{w_{2,0}}} - \frac{\eta}{4+\varepsilon}(-4)$

this descent roughly moves in the $\begin{bmatrix}1\\1\end{bmatrix}$ direction, which is different from before, $\left(\begin{bmatrix}2\\1\end{bmatrix}\right)$

So, no longer a minimum since it's not perpendicular to the empirical minimizer line.

1.2.2

No, both RMSprop and Adam try to rescale the gradient to have norm 1 on average to improve the slow update problem in low curvature dimension.

2. Gradient-based Hyper-parameter

2.1 Computation Graph of Learning Rates.

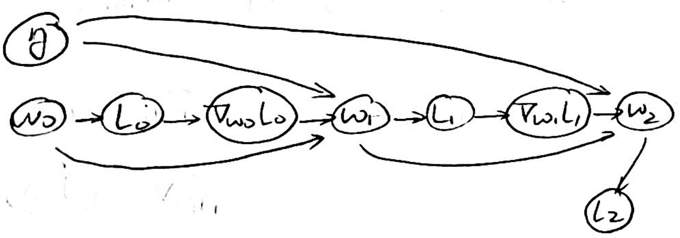
2.1.1

$w_1 = w_0 - \eta \nabla_{w_0} L_0$, $\nabla_{w_0} L_0 = \frac{2}{n} X^T (X\hat{w}_0 - t)$.

$w_2 = w_1 - \eta \nabla_{w_1} L_1$, $\nabla_{w_1} L_1 = \frac{2}{n} X^T (X\hat{w}_1 - t)$.

$L_2 = \frac{1}{n} \| X w_2 - t \|_2^2$,

Computation graph:



2.1.2

memory complexity for:

forward-propagation: $O(1)$.

back-propagation: $O(t)$.

t of such connections, all need to be stored in memory.

2.1.3.

Since hyper-parameters are shared by all iterations, $\frac{\partial w_i}{\partial \eta}$ need to be stored for all previous iterations. For realistic training with many iterations, we can easily run out of memory.

2.2 Learning Learning Rate.

$w_1 = w_0 - \eta \nabla_{w_0} L_0$.

2.2.1

$w_1 = w_0 - \eta \frac{2}{n} X^T (X w_0 - t)$.

$L_1 = \frac{1}{n} \| X w_1 - t \|_2^2$

$= \frac{1}{n} \| X(w_0 - \eta \frac{2}{n} X^T (X w_0 - t)) - t \|_2^2$. let $\underset{n\times 1}{a_0} = (X w_0 - t)$.

$= \frac{1}{n} \| X(w_0 - \eta \frac{2}{n} X^T a_0) - t \|_2^2$

2.2.2

$\frac{\partial L_1}{\partial \eta} = \frac{\partial}{\partial \eta} \left[\frac{1}{n} \| X w_0 - \eta \frac{2}{n} X X^T a_0 - t \|_2^2 \right]$.

$= (\underbrace{\frac{4}{n^2} X X^T a_0}_{1\times n})^T \underset{n\times 1}{(X w_0 - \eta \frac{2}{n} X X^T a_0 - t)}$.

$\underset{1\times 1}{\frac{\partial^2 L_1}{\partial \eta^2}} = \frac{4}{n^2} a_0^T X X^T \frac{2}{n} X X^T a_0$

$= \frac{8}{n^3} (X X^T a_0)^T (X X^T a_0) > 0$.

Therefore, L_1 is convex w.r.t η.

2.2.3

$\frac{\partial L_1}{\partial \eta} = 0$

$(a_0^T X X^T)(X w_0 - \eta^* \frac{2}{n} X X^T a_0 - t) = 0$.

$a_0^T X X^T X w_0 - \eta^* \frac{2}{n} a_0^T X X^T X X^T a_0 - a_0^T X X^T t = 0$

$\eta^* \frac{2}{n} \underbrace{a_0^T X X^T X X^T a_0}_{1\times 1} = a_0^T X X^T X w_0 - a_0^T X X^T t$

$\eta^* \frac{2}{n} = \frac{a_0^T X X^T X w_0 - a_0^T X X^T t}{a_0^T X X^T X X^T a_0}$

$\eta^* = \frac{n}{2} \cdot \frac{a_0^T X X^T X w_0 - a_0^T X X^T t}{a_0^T X X^T X X^T a_0}$.

check for denominator $= 0$,

when $X X^T a_0 = 0$,

$\frac{\partial L_1}{\partial \eta} = 0$, already at optimum, trivial.

3. Convolutional Neural Networks.

3.1. Convolutional filters.

Padding width $= \frac{k-1}{2} = \frac{3-1}{2} = 1$.

$$I = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad J = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}. \qquad I * J = \begin{bmatrix} -1 & 2 & 2 & -2 & 0 \\ -2 & 1 & 0 & 2 & -1 \\ 3 & 0 & 0 & 1 & -1 \\ -2 & 2 & 0 & 2 & -1 \\ 0 & -2 & 3 & -2 & 0 \end{bmatrix}$$

3.2. Size of ConvNets.

RGB image. $112 \times 112 \times 3$.
kernel size 3×3.

<u>layer</u>	<u># params.</u>
conv3-64	$64 \times (3 \times 3 \times 3 + 1)$. ✓ $= 1792$
max pool	0 $56 \times 56 \times 64$.
conv3-128	$128 \times (3 \times 3 \times 64 + 1)$. ✓ $= 73856$
max pool	0 $28 \times 28 \times 128$
conv3-256	$256 \times (3 \times 3 \times 128 + 1)$ ✓ $= 295168$
conv3-256	$256 \times (3 \times 3 \times 256 + 1)$. ✓ $= 590080$
max pool	0 $14 \times 14 \times 256$
fc-1024	$(14 \times 14 \times 256) \times 1024$. $= 51380224 + 1024$.
fc-100	$1024 \times 100 + 100$
softmax	0
total.	$\boxed{52493520} + 1124 = 52494644$.