Question 1 Optimization.

(a) Stochastic Gradient Descent (SGD).

1.1.1 Minimum Norm Solution. Show that SGD solution is identical to the minimum norm solution w^* obtained by gradient descent, i.e., $\hat{w} = w^*$.

The unique minimum norm solution w^*, obtained by gradient descent, is

$$w^* = X^T(XX^T)^{-1}t$$

We just need to show that the SGD converged solution $\hat{w}$ is identical to w^*, i.e.,

$$\hat{w} = X^T(XX^T)^{-1}t$$

As we can see,

$$\begin{aligned}
\mathcal{L} &= \frac{1}{n}||X\hat{w} - t||_2^2 \\
\mathcal{L}_i(x_i, w) &= ||\hat{w}^T x_i - t_i||^2 \\
\frac{\partial \mathcal{L}_i}{\partial \hat{w}} &= 2(\hat{w}^T x_i - t_i)x_i \\
\hat{w}_{t+1} &\leftarrow \hat{w}_t - 2\eta({\hat{w}_t}^T x_i - t_i)x_i
\end{aligned}$$

Since x_i is $d \times 1$, x_i can be written as $x_i = X^T \mathit{1}$, where X^T is $d \times n$ and $\mathit{1}$ is $n \times 1$.
Given i, $\mathit{1}^T$ will become [0 ... 1 ... 0], where the i_{th} entry is 1.
Thus, we have

$$\hat{w}_{t+1} \leftarrow \hat{w}_t - 2\eta X^T \mathit{1}({\hat{w}_t}^T x_i - t_i)$$

Then, we can use the same proof method applied on last homework to show that if $w_0 = 0$, we have

$$\hat{w} \propto X^T d$$

where d is $n \times 1$ vector.
To show SGD from zero initialization finds a minimum norm unique minimizer:

$$\begin{aligned}
X\hat{w} &= t \\
XX^T d &= t \\
d &= (XX^T)^{-1}t \\
\hat{w} &= X^T(XX^T)^{-1}t
\end{aligned}$$

Question 2 Gradient-based Hyper-parameter Optimization.

(a) Computation Graph of Learning Rates.

2.1.1 *Draw the computation graph.*

The computation graph is shown as Figure 1.

2.1.2 *What is the memory complexity for the forward-propagation to compute $\mathcal{L}_t$ in terms of t? What is the*

Question 2 [(a)] continued on next page. . .

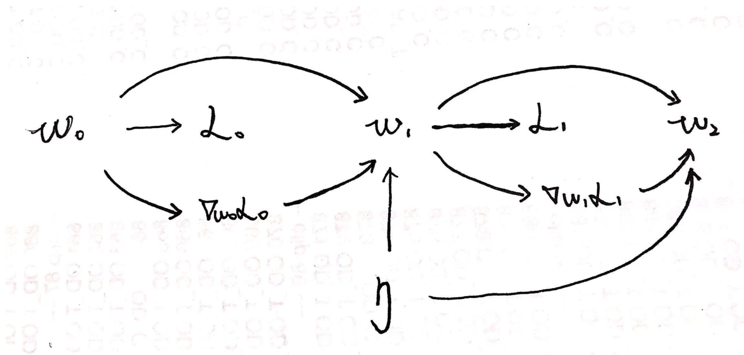

Figure 1: Computation graph.

memory complexity for using the standard back-propagation to compute the gradient w.r.t. the learning rate, $\nabla_\eta \mathcal{L}_t$ in terms of t?

To compute $\mathcal{L}_t$, we need to store w_t; to compute w_t, we need to store w_{t-1} and $\nabla_{w_{t-1}}\mathcal{L}_{t-1}$.
The shape of w_{t-1} and $\nabla_{w_{t-1}}\mathcal{L}_{t-1}$ are both $d \times 1$.
Thus, the memory complexity for the forward-propagation to compute $\mathcal{L}_t$ is O(d). In terms of t, the memory complexity for the forward-propagation is O(1).

$$\overline{\nabla_\eta \mathcal{L}_{t-1}} = \eta \overline{w_t}$$
$$\overline{w_t} = \overline{w_{t+1}} + \frac{2}{n} X^T X \overline{\nabla_\eta \mathcal{L}_t}$$

The shape of $\overline{w_t}$ and $\overline{\nabla_\eta \mathcal{L}_t}$ are both $d \times 1$.
Thus, the memory complexity for using the standard back-propagation to compute a specific gradient is O(d). In terms of t, the memory complexity for using the standard back-propagation to compute a specific gradient is O(1).
For computing all the gradients and **saving** them, the memory complexity is O(t).

2.1.3 *Explain one potential problem for applying gradient-based hyper-parameter optimization in more realistic examples where models often take many iterations to converge.*

One potential problem can be that for a realistic example where models often take many iterations to converge, if the optimized learning rate η_t becomes less with the increase of iteration t, the decreasing rate of $\mathcal{L}$ will also become less, which will drive the model for more iterations, i.e., take more time to converge.

(b) Learning Learning Rates.

2.2.1 *Write down the expression of w_1 in terms of w_0, η, t and X. Then, using the expression to derive the loss $\mathcal{L}_1$ after single GD iteration in terms of η.*

We have

$$\frac{\partial \mathcal{L}}{\partial w} = \frac{2}{n} X^T (Xw - t)$$
$$w_1 \leftarrow w_0 - \frac{2\eta}{n} X^T (Xw_0 - t) = w_0 - \frac{2\eta}{n} X^T a$$
$$\mathcal{L}_1 = \frac{1}{n} ||Xw_1 - t||_2^2 = \frac{1}{n} ||a - \frac{2\eta}{n} XX^T a||_2^2$$

2.2.2 *Determine if this $\mathcal{L}_1$ is convex w.r.t. the learning rate η.*

$$\begin{aligned}
\frac{\partial \mathcal{L}_1}{\partial \eta} &= \frac{4}{n^2} a^T XX^T (a - \frac{2\eta}{n} XX^T a) \\
&= \frac{4}{n^2} a^T XX^T a - \frac{8\eta}{n^3} a^T XX^T XX^T a \\
\frac{\partial \mathcal{L}_1^2}{\partial^2 \eta} &= \frac{8}{n^3} a^T XX^T XX^T a = \frac{8}{n^3} ||XX^T a||^2
\end{aligned}$$

As the second order derivative of $\mathcal{L}$ is positive, $\mathcal{L}_1$ is convex w.r.t. the learning rate η.

2.2.3 *Write down the derivative of $\mathcal{L}_1$ w.r.t. η and use it to find the optimal learning rate η^* that minimizes the loss after one GD iteration.*

$$\begin{aligned}
\frac{\partial \mathcal{L}_1}{\partial \eta} &= 0 \\
\frac{4}{n^2} a^T XX^T a - \frac{8\eta}{n^3} a^T XX^T XX^T a &= 0 \\
\eta^* &= \frac{n a^T XX^T a}{2 a^T XX^T XX^T a}
\end{aligned}$$

Question 3 Convolutional Neural Networks.

(a) Convolutional Filters. *Write down the values of the resulting matrix. What feature does this convolutional filter detect?*

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

(b) Size of ConvNets. *Calculate the number of parameters for this conv net including the bias units.*

Layers	Output Units	Parameters
Image	112*112*3	-
Conv3-64	112*112*64	3*3*3*64+64=1,792
MaxPool	56*56*64	-
Conv3-128	56*56*128	64*3*3*128+128=73,856
MaxPool	28*28*128	-
Conv3-256	28*28*256	128*3*3*256+256=295,168
Conv3-256	28*28*256	256*3*3*256+256=590,080
MaxPool	14*14*256	-
FC-1024	1024	14*14*256*1024+1024=51,381,248
FC-100	100	1024*100+100=102,500
Soft-max	100	-
Total	-	52,444,644