CSC413: Homework 2

Tianyu Du (1003801647)

2020/02/07 at 01:08:49

1 Optimization

1.1 Stochastic Gradient Descent (SGD)

1.1.1 Minimum Norm Solution

Answer. Recall from the previous homework that the solution found by gradient descent in the over-parameterized situation was

$$\mathbf{w}^* = X^T(XX^T)^{-1}\mathbf{t} \tag{1.1}$$

Moreover, the solution $\mathbf{w}^*$ reached by gradient descent was in the row space of X (i.e., the span of rows of X), and $\mathbf{w}^*$ is a zero loss solution.
For one single updating using $\mathbf{x}_i$ in the t^{th} iteration in the SGD process, the gradient is

$$\nabla_{\mathbf{w}_t}\mathcal{L}_i = \frac{\partial}{\partial \mathbf{w}_t}\left\|\mathbf{w}_t^T\mathbf{x}_i - t_i\right\|_2^2 \tag{1.2}$$

$$= \underbrace{2(\mathbf{w}_t^T\mathbf{x}_i - t_i)}_{\in\mathbb{R}}\mathbf{x}_i \tag{1.3}$$

$$\implies -\eta\nabla_{\mathbf{w}_t}\mathcal{L}_i \in Row(X) \tag{1.4}$$

Provided that the starting point $\mathbf{w}_0 = \mathbf{0} \in Row(X)$, for every t, $\mathbf{w}_t \in Row(X)$ by an inductive argument.
Because $\mathbf{w}_t \in Row(X)$, let

$$\mathbf{w}_t = X^T\mathbf{r}_t \quad \mathbf{r}_t \in \mathbb{R}^n \tag{1.5}$$

$$\hat{\mathbf{w}} = X^T\hat{\mathbf{r}} \quad \hat{\mathbf{r}} \in \mathbb{R}^n \tag{1.6}$$

Note that, if $\mathbf{x}_i$ is chosen for the t^{th} iteration of SGD, then only the i^{th} components of $\mathbf{r}_t$ and $\mathbf{r}_{t+1}$ are different, and all other components are the same.
Suppose the solution $\hat{\mathbf{w}}$ reached by SGD is a zero loss solution, that is, $X\hat{\mathbf{w}} = \mathbf{t}$. Then

$$XX^T\hat{\mathbf{r}} = \mathbf{t} \tag{1.7}$$

Because $\hat{\mathbf{w}}$ is a global minimum and SGD converges to this point, then it must be the case that

$$\nabla_{\hat{\mathbf{w}}}\mathcal{L}_i(\mathbf{x}_i, \hat{\mathbf{w}}) = 0 \quad \forall i \in \{1, \cdots, n\} \tag{1.8}$$

That is, one additional iteration of SGD does not improve performance no matter which sample is chosen. This is the same as

$$
\begin{aligned}
& \nabla_{r_i} \left\| XX^T\hat{\mathbf{r}} - \mathbf{t} \right\|_2^2 = 0 \quad \forall i \in \{1, \cdots, n\} && (1.9) \\
\iff & \nabla_{\mathbf{r}} \left\| XX^T\hat{\mathbf{r}} - \mathbf{t} \right\|_2^2 = 0 && (1.10) \\
\implies & (XX^T\hat{\mathbf{r}} - \mathbf{t})^T XX^T = 0 && (1.11) \\
\implies & XX^T(XX^T\hat{\mathbf{r}} - \mathbf{t}) = 0^T && (1.12) \\
\implies & XX^TXX^T\hat{\mathbf{r}} = XX^T\mathbf{t} && (1.13) \\
\implies & \hat{\mathbf{r}} = (XX^T)^{-1}\mathbf{t} && (1.14) \\
\implies & \hat{w} = X^T\hat{\mathbf{r}} = X^T(XX^T)^{-1}\mathbf{t} && (1.15) \\
& = \mathbf{w}^* && (1.16)
\end{aligned}
$$

■

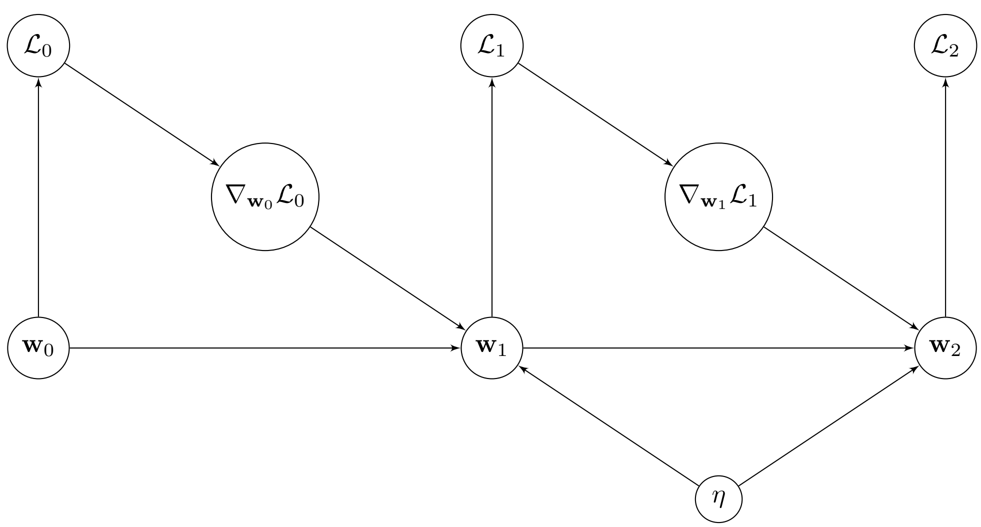
1.1.2 Mini-batch SGD (Optional)

Answer. ■

2 Gradient-Based Hyper-Parameter Optimization

2.1 Computation Graph of Learning Rates

2.1.1



2.1.2

Answer.
(**Forward-Propagation**) For each iteration t, the prediction $X\hat{\mathbf{w}} \in \mathbb{R}^n$ and label $\mathbf{t}$ take memory of size $2t$, the loss takes 1 unit of memory. Therefore, the overall memory complexity is $\mathcal{O}(d)$.

(**Back-Propagation**) $\nabla_\eta \mathcal{L}_t$ needs to be computed using chain rule through $\mathbf{w}_\tau$ for every $\tau \leq t$. Note that for every τ, $\nabla_{\mathbf{w}_\tau} \mathcal{L}_\tau \in \mathbb{R}^d$ and $\frac{d}{d\eta}\mathbf{w}_\tau \in \mathbb{R}^d$. The memory complexity is $\mathcal{O}(dt)$. ■

2.1.3

Answer. The memory complexity analysis in the previous question suggests the required memory of tracing gradient with respect to η grows linearly in the number of iterations, the memory cost can be prohibitive. ■

2.2 Learning Learning Rates

2.2.1

Answer.

$$\begin{aligned}
\mathbf{w}_1 &= \mathbf{w}_0 - \eta \nabla_{\mathbf{w}_0} \mathcal{L}_0 && (2.1)\\
&= \mathbf{w}_0 - \frac{2\eta}{n} X^T (X\mathbf{w}_0 - \mathbf{t}) && (2.2)\\
\mathcal{L}_1 &= \frac{1}{n} \left\| X \left[\mathbf{w}_0 - \frac{2\eta}{n} X^T (X\mathbf{w}_0 - \mathbf{t}) \right] - \mathbf{t} \right\|_2^2 && (2.3)\\
&= \frac{1}{n} \left\| X \left[\mathbf{w}_0 - \frac{2\eta}{n} X^T \mathbf{a}) \right] - \mathbf{t} \right\|_2^2 && (2.4)\\
&= \frac{1}{n} \left\| X\mathbf{w}_0 - \frac{2\eta}{n} X X^T \mathbf{a} - \mathbf{t} \right\|_2^2 && (2.5)\\
&= \frac{1}{n} \left\| \mathbf{a} - \frac{2\eta}{n} X X^T \mathbf{a} \right\|_2^2 && (2.6)
\end{aligned}$$

■

2.2.2

Answer. Note that the an arbitrary norm is convex: let $\lambda \in [0,1]$

$$\begin{aligned}
\|\lambda x + (1-\lambda) y\|_p &\leq \|\lambda x\|_p + \|(1-\lambda) y\|_p \text{ (triangle inequality)} && (2.7)\\
&= |\lambda| \, \|x\|_p + |1-\lambda| \, \|y\|_p && (2.8)\\
&= \lambda \|x\|_p + (1-\lambda) \|y\|_p && (2.9)
\end{aligned}$$

And $f(x) = x^2$ is strictly convex as well. Moreover, $\mathbf{a} - \frac{2\eta}{n} X X^T \mathbf{a}$ is linear in η. $\mathcal{L}_1(\eta)$ is a composite of linear, convex and strictly convex functions. Therefore, $\mathcal{L}_1(\eta)$ is strictly convex with respect to η. ■

2.2.3

Answer. The derivative of $\mathcal{L}_1$ w.r.t. η is

$$\nabla_\eta \mathcal{L}_1 = \nabla_\eta \frac{1}{n} \left\| \mathbf{a} - \frac{2\eta}{n} X X^T \mathbf{a} \right\|_2^2 \tag{2.10}$$

$$= \frac{2}{n} \left(\mathbf{a} - \frac{2\eta}{n} X X^T \mathbf{a} \right)^T X X^T \mathbf{a} \tag{2.11}$$

$$= \frac{2}{n} \left(\mathbf{a}^T - \frac{2\eta}{n} \mathbf{a}^T X X^T \right) X X^T \mathbf{a} \tag{2.12}$$

$$= \frac{2}{n} \left(\mathbf{a}^T X X^T \mathbf{a} - \frac{2\eta}{n} \mathbf{a}^T X X^T X X^T \mathbf{a} \right) \tag{2.13}$$

Set derivative to zero and solve for the optimal learning rate η^*:

$$\nabla_\eta \mathcal{L}_1 = 0 \tag{2.14}$$

$$\implies \frac{2}{n} \left(\mathbf{a}^T X X^T \mathbf{a} - \frac{2\eta}{n} \mathbf{a}^T X X^T X X^T \mathbf{a} \right) = 0 \tag{2.15}$$

$$\implies \left\| X^T \mathbf{a} \right\|_2^2 = \frac{2\eta^*}{n} \left\| X X^T \mathbf{a} \right\|_2^2 \tag{2.16}$$

$$\implies \eta^* = \frac{n \left\| X^T \mathbf{a} \right\|_2^2}{2 \left\| X X^T \mathbf{a} \right\|_2^2} \tag{2.17}$$

■

3 Convolutional Neutral Networks

3.1 Convolutional Filters

Answer.

$$\mathbf{I} * \mathbf{J} = \begin{bmatrix} -1 & 2 & 2 & -2 & 0 \\ -2 & 1 & 0 & 2 & -1 \\ 3 & 0 & 0 & 1 & -1 \\ -2 & 2 & 0 & 2 & -1 \\ 0 & -2 & 3 & -2 & 0 \end{bmatrix} \tag{3.1}$$

This convolutional filter detect edges. ■

3.2 Size of ConvNets

Answer. The table below presents numbers of parameters in all layers.

Layer	Dim. In	Dim. Out	Num. Weights	Num. Bias	Total Params.
Conv3-64	112*112*3	112*112*64	3*3*3*64	64	1,792
Max Pool	112*112*64	56*56*64	0	0	0
Conv3-128	56*56*64	56*56*128	3*3*64*128	128	73,856
Max Pool	56*56*128	28*28*128	0	0	0
Conv3-256	28*28*128	28*28*256	3*3*128*256	256	295,168
Conv3-256	28*28*256	28*28*256	3*3*256*256	256	590,080
Max Pool	28*28*256	14*14*256	0	0	0
FC-1024	14*14*256	1024	14*14*256*1024	1024	51,381,248
FC-100	1024	100	1024*100	100	102,500
Softmax	100	100	0	0	0
Total					52,444,644

■