Custom Neural Networks

1 Vanilla 2-layers NN

Forward

$X \in \mathcal{R}^{n\times m}$, $W_1 \in \mathcal{R}^{d\times m}$, $W_2 \in \mathcal{R}^{k\times dn}$, $B_1 \in \mathcal{R}^{d\times n}$, $B_2 \in \mathcal{R}^{k}$, $y \in \mathcal{R}^{k}$

$$
\begin{aligned}
pre_{a_1} &= \frac{X - \mathbb{E}(X)}{\sqrt{\mathbb{E}(X - \mathbb{E}X)^2}} && \in \mathcal{R}^{n\times m} && (1)\\
a_1 &= W_1 pre_{a_1}^{\intercal} + B_1 && \in \mathcal{R}^{d\times n} && (2)\\
h_1 &= \texttt{ReLU}(a_1) && \in \mathcal{R}^{d\times n} && (3)\\
pre_{a_2} &= Flatten(h_1) && \in \mathcal{R}^{dn} && (4)\\
a_2 &= (W2pre_{a_2}^{\intercal})^{\intercal} + B_2 && \in \mathcal{R}^{k} && (5)\\
h_2 &= a_2 I_k && \in \mathcal{R}^{k} && (6)\\
s &= Softmax(h_2) && \in \mathcal{R}^{k} && (7)\\
\mathcal{L} &= -(y \odot \ln s)\mathbf{1}_k^{\intercal} && \in \mathcal{R} && (8)
\end{aligned}
$$

$$\mathcal{L}(X, W_1, W_2, y) = CrossEntropy(y, Softmax(W_2 Flatten(\texttt{ReLU}(W_1 X + B1)) + B2))$$

Backward

We have

$$\left(\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_2}\right)_{k\times dn} = \left(\frac{\partial \mathcal{L}}{\partial s}\right)_k \left(\frac{\partial s}{\partial h_2}\right)_{k\times k} \left(\frac{\partial h_2}{\partial a_2}\right)_{k\times k} \left(\frac{\partial a_2}{\partial W_2}\right)_{k\times k\times dn} \quad (1)$$

or, equivalently,

$$\left(\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_2}\right)_{k\times dn} = \left(\frac{\partial \mathcal{L}}{\partial h_2}\right)_k \left(\frac{\partial h_2}{\partial a_2}\right)_{k\times k} \left(\frac{\partial a_2}{\partial W_2}\right)_{k\times k\times dn} \quad (2)$$

In a simplified vectorized form, we have

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_2} = ((\boldsymbol{s}-\boldsymbol{y}) \odot \mathbf{1}_k)^\intercal \boldsymbol{pre_{a_2}} \tag{3}$$

Meanwhile

$$\begin{aligned}\left(\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_1}\right)_{d\times m} = &\left(\frac{\partial \mathcal{L}}{\partial h_2}\right)_k \left(\frac{\partial h_2}{\partial a_2}\right)_{k\times k} \left(\frac{\partial a_2}{\partial pre_{a_2}}\right)_{k\times dn} \left(\frac{\partial pre_{a_2}}{\partial h_1}\right)_{dn\times d\times n} \\ &\left(\frac{\partial h_1}{\partial a_1}\right)_{d\times n\times d\times n} \left(\frac{\partial a_1}{\partial W_1}\right)_{d\times n\times d\times m}\end{aligned} \tag{4}$$

Therefore,

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_1} = ((\boldsymbol{s}-\boldsymbol{y}) \odot \mathbf{1}_k \boldsymbol{W_2})_{dn\to d\times n} \odot \left(\frac{\partial \mathtt{ReLU}(\boldsymbol{a_1})}{\partial \boldsymbol{a_1}}\right)_{d\times n} \boldsymbol{pre_{a_1}} \tag{5}$$

Biases gradients are computed as follows:

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}B_2} = (\boldsymbol{s}-\boldsymbol{y}) \odot \mathbf{1}_k \tag{6}$$

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}B_1} = ((\boldsymbol{s}-\boldsymbol{y}) \odot \mathbf{1}_k \boldsymbol{W_2})_{dn\to d\times n} \odot \left(\frac{\partial \mathtt{ReLU}(\boldsymbol{a_1})}{\partial \boldsymbol{a_1}}\right)_{d\times n} \tag{7}$$

2 1-layer Transformer Encoder

Forward

$X \in \mathcal{R}^{n \times d_0}$, $W_1 \in \mathcal{R}^{d_0 \times d_1}$, $\texttt{cls\_tok} \in \mathcal{R}^{1 \times d_1}$, $W_{\{q,k,v\}} \in \mathcal{R}^{d_1 \times d_2}$, $W_2 \in \mathcal{R}^{d_2 \times K}$, $y \in \mathcal{R}^K$

$$
\begin{aligned}
h_0 &= \frac{X - \mathbb{E}(X)}{\sqrt{\mathbb{E}(X - \mathbb{E}X)^2}} && \in \mathcal{R}^{n \times d_0} && (1)\\
h_1 &= h_0 W_1 && \in \mathcal{R}^{n \times d_1} && (2)\\
h_{pre_2} &= \texttt{concat}(h_1, \texttt{cls\_tok}) && \in \mathcal{R}^{(n+1) \times d_1} && (3)\\
\{Q, K, V\} &= h_{pre_2} W_{\{q,k,v\}} && \in \mathcal{R}^{(n+1) \times d_2} && (4)\\
S &= Softmax(QK^\intercal) && \in \mathcal{R}^{(n+1) \times (n+1)} && (5)\\
h_2 &= SV + h_{pre_2} && \in \mathcal{R}^{(n+1) \times d_2} && (6)\\
z &= h_{2_{(n+1)}} W2 && \in \mathcal{R}^K && (7)\\
s &= Softmax(z) && \in \mathcal{R}^K && (8)\\
\mathcal{L} &= -\frac{1}{K} \mathbf{1}_{d_2} (y \odot \ln s)^\intercal && \in \mathcal{R} && (9)
\end{aligned}
$$

$\mathcal{L}(X, W_1, \texttt{cls\_tok}, W_{\{q,k,v\}}, W_2, y) =$
$CrossEntropy(y, Softmax(SelfAttention(\texttt{concat}((XW_1), \texttt{cls\_tok}), W_{\{q,k,v\}})_{\texttt{[seq\_len+1,:]}} W2)))$

Backward

$$
\begin{aligned}
\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_2} &= \left((\boldsymbol{z} - \boldsymbol{y})^\intercal \boldsymbol{h_{2_{(n+1)}}}\right)^\intercal = \boldsymbol{h}^\intercal_{\boldsymbol{2}_{(n+1)}} (\boldsymbol{z} - \boldsymbol{y}) && (1)\\
\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_v} &= \left(((\boldsymbol{z} - \boldsymbol{y}) \boldsymbol{W}_{\boldsymbol{2}}^\intercal)^\intercal \boldsymbol{S_{(n+1)}} \boldsymbol{h_{pre_2}}\right)^\intercal = \boldsymbol{h}^\intercal_{\boldsymbol{pre_2}} \boldsymbol{S}^\intercal_{\boldsymbol{(n+1)}} (\boldsymbol{z} - \boldsymbol{y}) \boldsymbol{W}_{\boldsymbol{2}}^\intercal && (2)
\end{aligned}
$$

Now, with

$$
\begin{aligned}
\boldsymbol{\alpha} &= diag(\boldsymbol{S_{(n+1)}}) - (\boldsymbol{Q_{(n+1)}} \boldsymbol{K}^\intercal)^\intercal (\boldsymbol{Q_{(n+1)}} \boldsymbol{K}^\intercal) && (3)\\
&= diag(\boldsymbol{S_{(n+1)}}) - \boldsymbol{K} \boldsymbol{Q}^\intercal_{\boldsymbol{(n+1)}} \boldsymbol{Q_{(n+1)}} \boldsymbol{K}^\intercal && (4)\\
&= \frac{\partial \boldsymbol{S_{(n+1)}}}{\partial \boldsymbol{Q_{(n+1)}} \boldsymbol{K}^\intercal} && (5)
\end{aligned}
$$

we have

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_k} = \big((((\boldsymbol{z}-\boldsymbol{y})\boldsymbol{W_2^\intercal})^\intercal \boldsymbol{V}^\intercal \boldsymbol{\alpha})^\intercal \boldsymbol{Q_{(n+1)}} \boldsymbol{h_{pre_2}}\big)^\intercal \tag{6}$$

$$= \boldsymbol{h_{pre_2}^\intercal} \boldsymbol{Q_{(n+1)}^\intercal} ((\boldsymbol{z}-\boldsymbol{y})\boldsymbol{W_2^\intercal})^\intercal \boldsymbol{V}^\intercal \boldsymbol{\alpha} \tag{7}$$

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_q} = \Big(((((\boldsymbol{z}-\boldsymbol{y})\boldsymbol{W_2^\intercal})^\intercal \boldsymbol{V}^\intercal \boldsymbol{\alpha} \boldsymbol{K})^\intercal \boldsymbol{h_{pre_{2_{(n+1)}}}}\Big)^\intercal \tag{8}$$

$$= \boldsymbol{h_{pre_{2_{(n+1)}}}^\intercal} ((\boldsymbol{z}-\boldsymbol{y})\boldsymbol{W_2^\intercal})^\intercal \boldsymbol{V}^\intercal \boldsymbol{\alpha} \boldsymbol{K} \tag{9}$$

And with

$$\boldsymbol{\beta} = \boldsymbol{Q_{(n+1)}} \boldsymbol{W_k^\intercal} + \boldsymbol{W_q} \boldsymbol{K}^\intercal \tag{10}$$

we have

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\texttt{cls\_tok}} = \big(((\boldsymbol{z}-\boldsymbol{y})\boldsymbol{W_2^\intercal})^\intercal (\boldsymbol{S}_{(n+1)} \boldsymbol{W}_v + \boldsymbol{\alpha}\boldsymbol{\beta}\boldsymbol{V} + \boldsymbol{1})\big)^\intercal \tag{11}$$

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}W_1} = \big(((\boldsymbol{z}-\boldsymbol{y})\boldsymbol{W_2^\intercal})^\intercal (\boldsymbol{S}_{(n+1)} \boldsymbol{W}_v \boldsymbol{h}_0 + \boldsymbol{\alpha}\boldsymbol{\beta}\boldsymbol{h}_0\boldsymbol{V} + \boldsymbol{h}_0)\big)^\intercal \tag{12}$$