Hw 9 (3039725444)

Chen Yuanteng

1. Self-Supervised Linear Purification

(a).

α: $X - WX = 0$ Reconstruction Loss $= 0$

$\lambda \|W\|_F^2 = 2\lambda$ Regularization Loss $= 2\lambda$

β: $X - WX = \begin{bmatrix} -2.17 & 1.98 & 2.41 & -2.03 \\ 0.02 & -0.01 & 0.01 & -0.02 \end{bmatrix} - \begin{bmatrix} -2.17 & 1.98 & 2.41 & -2.03 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

$= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.02 & -0.01 & 0.01 & -0.02 \end{bmatrix}$

Reconstruction loss $= 0.001$

Regularization loss $= \lambda$

ii: when $L_2(\alpha) = L_2(\beta)$

$2\lambda = 0.001 + \lambda$

$\lambda = 0.001$

when $\lambda > 0.001$, $W^{(\alpha)}$ is not a better solution than $W^{(\beta)}$

(b) $L_2(W, X, \lambda) = \|X - WX\|_F^2 + \lambda \|W\|_F^2$

$W \in \mathbb{R}^{m \times m}$ $\sigma_1 > \cdots > \sigma_m \geq 0$ are the m singular values in X, $X = U\Sigma V^T$

(i).

$$\hat{W} = [U] \begin{bmatrix} \frac{\sigma_1^2}{\sigma_1^2+\lambda} & & & \\ & \frac{\sigma_2^2}{\sigma_2^2+\lambda} & & \\ & & \ddots & \\ & & & \frac{\sigma_m^2}{\sigma_m^2+\lambda} \end{bmatrix} [U^T]$$

(ii).

$$L_2(W; X, \lambda) = \|X - WX\|_F^2 + \lambda\|W\|_F^2$$

$$= \sum_{i=1}^m \|X_i - W_i X\|_2^2 + \lambda \|W_i\|_2^2 \quad (X_i \in R^{1\times n}, W_i \in R^{1\times m})$$

$$= \sum_{i=1}^m \|X_i^T - X^T W_i^T\|_2^2 + \lambda \|W_i\|_2^2$$

$$\Updownarrow$$

$$\|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2 \quad \hat{\beta} = (X^T X + \lambda I)^{-1} X^T y$$

$$\therefore \hat{W}_i^T = (XX^T + \lambda I)^{-1} X X_i^T$$

$$\hat{W}^T = (XX^T + \lambda I)^{-1} X X^T$$

$$\hat{W} = XX^T (XX^T + \lambda I)^{-1}$$

plug in $X = U\Sigma V^T$

$$\hat{W} = U\Sigma V^T (U\Sigma V^T)^T (U\Sigma V^T (U\Sigma V^T)^T + \lambda I)^{-1}$$

$$= U\Sigma\Sigma^T U^T (U\Sigma\Sigma^T U^T + \lambda I)^{-1}$$

$$= U\Sigma\Sigma^T U^T U (\Sigma\Sigma^T + \lambda I)^{-1} U^T$$

$$= U\Sigma\Sigma^T (\Sigma\Sigma^T + \lambda I)^{-1} U^{-T}$$

(c) data matrix $X \in R^{8\times n}$ has the following singular value $\{\sigma_i\} = \{10, 8, 4, 1, 0.5, 0.36, 0.16, 0.01\}$

Solution:

① $\forall i \in \{1, 2, 3\}$ $\dfrac{\sigma_i^2}{\sigma_i^2 + \lambda} \geq 0.8$

$$\frac{10^2}{10^2+\lambda} > \frac{8^2}{8^2+\lambda} > \frac{4^2}{4^2+\lambda} \geq 0.8$$

$$\lambda \leq 4$$

② $\forall i \in \{4, 5, 6, 7, 8\}$ $\dfrac{\sigma_i^2}{\sigma_i^2 + \lambda} \leq 0.5$

$$0.5 \geq \frac{1^2}{1^2+\lambda} > \cdots > \frac{0.01^2}{0.01^2+\lambda}$$

$$\lambda \geq 1$$

$$\therefore \quad 1 \leq \lambda \leq 4$$

3. Argmax Attention

(a). $\langle [1,1,2]^T, [1,2,0]^T \rangle = 3$

$\langle [1,1,2]^T, [0,3,4]^T \rangle = 11$

$\langle [1,1,2]^T, [5,0,0]^T \rangle = 5$

$\langle [1,1,2]^T, [0,0,1]^T \rangle = 2$

$\mathrm{argmax}([3,11,5,2]) = [0,1,0,0]$

· output

$= 0\cdot[2,0,1]^T + 1\cdot[1,4,3]^T + 0\cdot[0,-1,4]^T + 0\cdot[1,0,-1]^T$

$= [1,4,3]^T$

(b). It wouldn't work in real life, since the gradients only flow through the one element that is selected by argmax and thus most of the parameters in the transformer layers would remain the same if we did gradient-based training.

→ the argmax is not sensitive to small changes in the keys and queries, since any such tiny perturbations will not change the winner.

Consequently, the gradients with respect to the keys and queries will always be zero.

→ the keys and queries will never be improved.

4. Justifying Scaled-Dot Product Attention

(a) Define $E[q^k k]$ in terms of μ, σ and d

Solution: $E(q^T k) = E(\sum_{i=1}^{d} q_i k_i)$

$= \sum_{i=1}^{d} E(q_i \cdot k_i) = \sum_{i=1}^{d} E(q_i) E(k_i) = \sum_{i=1}^{d} \mu_i^2 = \|\mu\|_2^2$

(b). $\mu = 0$, $\sigma = 1$, define $Var(q^T k)$ in terms of d.

Solution:

$Var(q^T k) = E[(q^T k)^2] - (E[q^T k])^2$

$E[(q^T k)^2] = E[\sum_{i=1}^{d} q_i^2 \cdot k_i^2] = \sum_{i=1}^{d} E(q_i^2 \cdot k_i^2)$

$= \sum_{i=1}^{d} E(q_i^2) \cdot E(k_i^2)$

$E(q_i^2) = E(k_i^2) = \mu^2 + \sigma^2$

$\therefore E[(q^T k)^2] = \sum_{i=1}^{d} [(\mu_i^2 + \sigma^2)(\mu_i^2 + \sigma^2)] = d\sigma^2$

$(E[q^T k])^2 = (\sum_{i=1}^{d} E(q_i) E(k_i))^2 = 0$

$\therefore Var(q^T k) = d \cdot \sigma^2 = d$.

(c) $\mu = 0$, $\sigma = 1$ Suppose we want $E(\frac{q^T k}{s}) = 0$

$Var(\frac{q^T \cdot k}{s}) = 1$. what should s be in terms of d.

Solution: as $\mu = 0$, $E(\frac{q^T k}{s}) = \frac{\|\mu\|_2^2}{s} = 0$

$Var(\frac{q^T \cdot k}{s}) = \frac{d}{s^2} = 1 \quad s = \sqrt{d}$

5. Kernelized Linear Attention.

$W_Q \in R^{F\times D}$. $W_K \in R^{F\times D}$ $W_V \in R^{F\times M}$.

$X \in R^{N\times F}$ (N-length sequence, F features.)

$Q = XW_Q \in R^{N\times D}$. $K = XW_K \in R^{N\times D}$

$V = XW_V \in R^{N\times M}$

$A_l(X) = V' = \text{softmax}(\frac{QK^T}{\sqrt{D}})V \in R^{N\times M}$

$T_l(X) = f_l(A_l(X) + X)$

$$V_i' = \frac{\sum_{j=1}^{N} sim(Q_i, K_j) V_j}{\sum_{j=1}^{N} sim(Q_i, K_j)}$$

(a). Solution,

when $\sum_{j=1}^{k} sim(Q_i, k_j) \neq 0$, V_i' can be finite

this means the function value of sim must have the same sign (all positive or negative).

(b).

(i) $K(x,y) = (\alpha x(x\cdot y) + c)^d$.

when using polynomical kernel attenetion:

$sim(q,k) = (q^T k + 1)^2$

(ii). assume $(q^T k + 1)^2 = \phi(q)^T \cdot \phi(k)$

$\phi(q) = [1, \sqrt{2}q_1, \cdots \sqrt{2}q_D, q_1^2, \cdots, q_D^2]^T$

$\phi(k) = [1, \sqrt{2}k_1, \cdots, \sqrt{2}k_D, k_1^2, \cdots, k_D^2]^T$

$$\phi(q)^T \cdot \phi(k) = 1 + 2q_1k_1 + \cdots + 2q_Dk_D + q_1^2k_1^2 + \cdots + q_D^2k_D^2$$
$$= \phi(q)^T \cdot \phi(k)$$

So $\phi(X) = [1, \sqrt{2}X_1, \sqrt{2}X_2, \cdots \sqrt{2}X_D, X_1^2, X_2^2, \cdots X_D^2]$

(iii) $K(q,k) = (\phi(q)^T \cdot \phi(k))$

$$V_i' = \frac{\sum_{j=1}^{N} sim(Q_i, k_j)V_j}{\sum_{j=1}^{N} sim(Q_i, k_j)} = \frac{\sum_{j=1}^{N} (\phi(Q_i)^T \cdot \phi(k_j)) \cdot V_j}{\sum_{j=1}^{N} (\phi(Q_i)^T \cdot \phi(k_j))}$$

c ( ) -

computational graph:

$$V_i' = \frac{\sum_{j=1}^{N} \exp\left(\frac{Q_i^T k_j}{\sqrt{D}}\right) V_j}{\sum_{j=1}^{N} \exp\left(\frac{Q_i^T k_j}{\sqrt{D}}\right)}$$

```
for i in range(N):
    for j in range(N):
        S[i,j] = Q[i, :].T @ K[j, :]/sqrt(D)

for i in range(N):
    Z = 0.
    for j in range(N):
        Z += exp(Z[i,j])
    for j in range(N):
        A[i,j] = exp(Z[i,j])/Z

for i in range(N):
    O[i, :] = 0
```

```
for j in range(N):
    O[i,:] += A[i,j] * V[j,:]
```

Time:

So there are three loop in the computational graph, the complexity are $O(N^2D)$, $O(N^2)$, $O(N^2M)$ respectively, so the total time complexity is $O(N^2 \max(D,M))$

Space:

$Q, K \in R^{N\times D}$, so they cost $O(ND)$ memory,

$S, A \in R^{N\times N}$, so they cost $O(N^2)$ memory

$V, O \in R^{N\times M}$, so they cost $O(NM)$ memory.

$\therefore$ total space complexity is $O(N \max(D,N,M))$

$$\text{complexity} = \begin{cases} O(N^2 \max(D,M)) & \text{time} \\ O(N \max(D,N,M)) & \text{space} \end{cases}$$

(d). $\mathrm{sim}(x,y) = K(x,y) = (\phi(Q_i)^T \phi(k_j))$

So. $V_i' = \dfrac{\sum_{j=1}^{N} \phi(Q_i)^T \phi(k_j) V_j}{\sum_{j=1}^{N} \phi(Q_i)^T \phi(k_j)} = \dfrac{\phi(Q_i)^T \sum_{j=1}^{N} \phi(k_j) V_j}{\phi(Q_i)^T \cdot \sum_{j=1}^{N} \phi(k_j)}$

Computational graph: $K \in R^{N\times D}$. $V \in R^{N\times M}$

U[:,:] = 0

for j in range(N):

$$U[:,:] \mathrel{+}= \overset{R^{C\times 1}}{\uparrow}\text{phi}(K[j,:]) \;@\; \overset{R^{1\times M}}{\uparrow}V[j,:].T$$

Z[:] = 0.

for j in range(N):

$$Z[:] \mathrel{+}= \overset{R^{C\times 1}}{\uparrow}\text{phi}(K[j,:]).$$

for j in range(N):

$$O[i,:] = \overset{R^{1\times C}}{\uparrow}\text{phi}(Q[i,:]).T \;@\; \overset{R^{C\times M}}{\uparrow}U[:,:] \rightarrow R^{1\times M}$$

$$O[i,:] \mathrel{/}= \text{phi}(Q[i,:]).T \;@\; Z[:]$$

$\therefore$ there are also 3 Loop in the computational graph

Time: time cost are $O(NCM)$. $O(NC)$, $O(NCM)$ $\therefore$ total time cost is $O(NCM)$

for quadratic kernel, $C = O(D^2)$

$\therefore$ time cost is $O(ND^2M)$ and in practice

$M \approx D$ $\therefore$ time cost is $O(ND^3)$

when $N \gg D^2$. $O(ND^3) \ll O(N^2D)$.

kernelized linear attention with a quadratic polynomical kernel is faster than softmax attention.

Space: $Q, K \to O(ND)$. memory.

$U \to O(CM)$ memory

$V, O \to O(NM)$ memory

$\therefore$ total space cost $\to O(ND + CM + NM)$

$= O(N(D+M) + CM)$

$$\xrightarrow{M \simeq D,\ C = D^2} O(ND + D^3)$$

so when $N >> D^2$, kernelized linear attention uses much less memory than softmax attention.

6. Kernelized Linear Attention (Part II)

(a). $K_{guass}(q,k) = \exp\left(\frac{-\|q-k\|_2^2}{2\sigma^2}\right)$

rewrite the softmax similarity function using Guassian Kernel:

$$\text{Sim}_{softmax}(q,k) = \exp\left(\frac{q^T k}{\sqrt{D}}\right)$$

$$q^T k = \frac{1}{2}\left(\|q\|_2^2 + \|k\|_2^2 - \|q-k\|_2^2\right)$$

$$\therefore \text{Sim}_{softmax}(q,k) = \exp\left(\frac{\|q\|_2^2}{2\sigma^2}\right) \cdot \exp\left(\frac{-\|q-k\|_2^2}{2\sigma^2}\right) \cdot \exp\left(\frac{\|k\|_2^2}{2\sigma^2}\right)$$

$$= \exp\left(\frac{\|q\|_2^2}{2\sigma^2}\right) \cdot K_{guass}(q,k) \cdot \exp\left(\frac{\|k\|_2^2}{2\sigma^2}\right)$$

记: $\phi_{random}(q) = \sqrt{\frac{1}{D_{random}}}\left[\sin(w_1 q), \cdots, \sin(w_{D_{random}} q), \cos(w_1 q), \cdots, \cos(w_{D_{random}} q)\right]^T$

D random of D-dimensional random vector W_i independently sampled from $N(0, \sigma^2 I_D)$

$E_{w_i}[\phi(q) \cdot \phi(k)] = \exp(-\|q-k\|^2 / 2\sigma^2)$

Solution:

$$\text{Sim}_{\text{softmax}}(q,k) = \exp\left(\frac{\|q\|_2^2}{2\sigma^2}\right) * K_{\text{gauss}}(q,k) * \exp\left(\frac{\|k\|_2^2}{2\sigma^2}\right)$$

$$= \exp\left(\frac{\|q\|_2^2}{2\sigma^2}\right) * \phi_{\text{random}}(q)^T \phi_{\text{random}}(k) * \exp\left(\frac{\|k\|_2^2}{2\sigma^2}\right)$$