hw 6

Chen Yuanteng

3039725444

1. Debugging DNNs.

(a)

potential reasons:

(1) Overfitting: 56-layer model is more complex and deeper, might be more prone to overfitting the training data, while the smaller 20-layer model may have better generalization capabilities.

(2) Vanishing or exploding gradients:
The deeper network architecture could lead to the problem of vanishing or exploding gradients.

(3). maybe training data is limited, it needs more data to train a model as deep and large as 56-layer model.

To mitigate this problem:

(1). Add regularization and dropout layer. to reduce the risk of overfitting.

(2) Add residual connections. which can aid information to flow in deeper network and alleviate the issue of vanishing or exploding gradient.

(b)

the model with layer normalization will not pass the test.

Because layer normalization computes the mean and var across the spatial dimensions for each individual sample in the batch.

In the provided gradient accumulation algorithm, the model accumulates gradients and updates the parameters every accumulated steps.

However, since layer normalization compute statistics per sample. the accumulated gradients from different samples within the same effective batch would have different mean and var values.

(c). in for loop, optimizer.zero_grad()

should be implemented after optimizer.step().

Otherwise, gradients would be accumulated

during every (inputs, label).

2. Tensor. Rematerialization.

(a) to compute activation of layer-9,
we need to compute activations of 6, 7, 8
first, so it needs 4 fwd in total
So to compute activations of 6, 7, 8, 9.
it needs 4+3+2+1 = 10, same as layer 1-4
∴ 2×10 = 20 fwd in total.

(b). when computing activations of 6, 7, 8, 9.
4 Loadmem are necessary,
to compute 1, 2, 3, 4. another 4 Loadmem
are needed. So 2×4 = 8 Loadmem in total.

(c)
during a single backward:
in tensor rematerialization.
20·2ms + 8·1ms = 48ms.
in storing all activations on this disk:
10·(time of Loaddisk) = 48ms.
time of Loaddisk = 48ms

3. Graph Dynamics

(a). $G_{t+1} = A \cdot G_t \quad G_0 = E$

$\therefore G_K = A^K$ and j-th node in G_K is the j-th row of G_K.

$\therefore$ the output of the j-th node at layer K in this network $\rightarrow A^K_j$

(b). using induction:

$L_0(i,j) = [i=j]$

$L_1(i,j)$ is the definition of matrix A

assume $L_h(i,j) = [A_h]_{i,j} \quad (h \geq 1)$

try to verify $L_{h+1}(i,j) = [A_{h+1}]_{i,j}$

from node i to node j with distance $= h+1$

$\Leftrightarrow$ from node i to node X with distance $= h$ + from node X to node j with an edge.

$\therefore L_{h+1}(i,j) = \sum_{X \in V(j)} L_h(i,X)$

$\therefore L_{h+1}(i,j) = \sum_{x=1}^{n} L_h(i,x) \cdot A_{x,j}$

$\because L_h(i,j) = [A_h]_{i,j}$.

$\therefore L_{h+1}(i,j) = \sum_{x=1}^{n} [A_h]_{i,x} A_{x,j} = [A_{h+1}]_{i,j}$

cc). in the matrix, each row $\Leftrightarrow$ each node

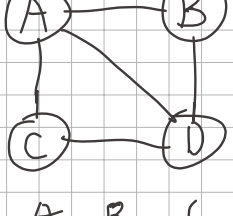
the graph is a set of nodes:

$$V_j = \sum_{i \in V(j)} \cdot V_i$$

cd:. replace sum aggregation with max aggregation:

outputs of node j at layer k $=$

- 1 — there is a path from i to j with length k
- 0 — otherwise



	A	B	C	D
A	0	1	1	1
B	1	0	0	1
C	1	0	0	1
D	1	1	1	0

$= A$

$$A^2 = \begin{matrix} 3. & 1 & 1 & 2. \\ 1 & 2 & 2 & 1 \\ 1 & 2 & 2 & 1 \\ 2 & 1 & 1 & 2 \end{matrix}$$