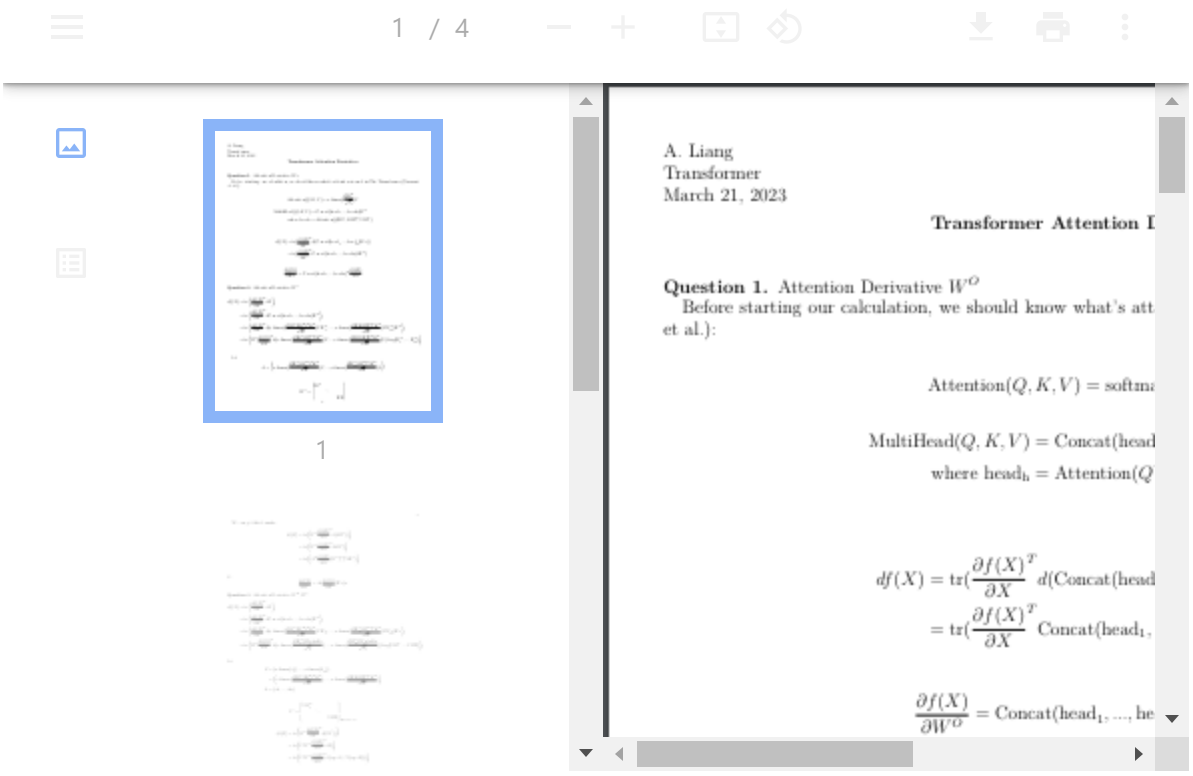
Written by Longx
on September 07, 2022

Transformer Attention Layer gradient

Transformer Attention Layer gradient.

The Transformer Attention Layer gradient



</embed>

Validate

This chapter will validate that my conclusion is true, so I code a python code with PyTorch, you can see it in https://github.com/Say-Hello2y/Transformer-attention.

Here I will select the code for validating W^Q, you can validate others in my https://github.com/Say-Hello2y/Transformer-attention.

Here is the code :

```
import torch
from multi_head_test import MultiHeadAttention



x =torch.rand(1,5,10)
attention = MultiHeadAttention(d_model=10, n_head=2)
```

```
out,gradient_wo,att1,score,A = attention(q=x, k=x, v=x)
'''
out 就是正常多头注意力的输出，gradient_wo是Wo的梯度,att1是未经过拼
就是未经过softmax的attention计算结果，跟推导中的定义一致

out is the output of the Multi_head attention,gradient_wo is t
after concat, this is because our batch_size  is 1, and A is t

'''
# print(A)
x=x.squeeze()
print('x is {}'.format(x))
I=torch.block_diag(torch.ones(5,5),torch.ones(5,5))
Y=1/(A.exp()@I+1e-15) # add a small positive to prevent divid
# print(attention.w_concat.weight)
vp = torch.block_diag(x@attention.w_v.weight.transpose(0,1)[:,
# print(vp)
dev_A = ((torch.ones(5,10)/50)@attention.w_concat.weight@vp.tr


# print(dev_A)
# print(attention.w_k.weight.transpose(0,1)[:,0:5])
ph1 = torch.cat((torch.eye(5,5),torch.zeros(5,5)),1)
ph2 = torch.cat((torch.zeros(5,5),torch.eye(5,5)),1)
w_q1 = (1/torch.sqrt(torch.tensor(5)))*x.transpose(0,1)@dev_A@
w_q2 = (1/torch.sqrt(torch.tensor(5)))*x.transpose(0,1)@dev_A@
w_q = torch.cat((w_q1,w_q2),1)
print('w_q_theory is {}'.format(w_q))
# print(out.mean())
print()
loss = out.mean()
loss.backward()
# loss=criterion(out, trg)
wq_gradient = attention.w_q.weight.grad
w_q_true = wq_gradient.transpose(0,1)
print('w_q_true is {}'.format(w_q_true))
print()
print('W^Q error is {}'.format((w_q-w_q_true).short()))
```

Here is the output:

```
x is tensor([[0.9828, 0.7096, 0.8925, 0.7485, 0.2119, 0.3197,
         0.7579],
        [0.6402, 0.9792, 0.2407, 0.1695, 0.2291, 0.5823, 0.095
         0.3585],
        [0.1755, 0.8609, 0.5094, 0.1457, 0.0729, 0.8283, 0.454
         0.8498],
        [0.7441, 0.0862, 0.0149, 0.2518, 0.9933, 0.9696, 0.680
         0.0737],
        [0.9417, 0.1756, 0.5198, 0.9735, 0.9840, 0.2650, 0.094
         0.8001]])
w_q_theory is tensor([[-3.3509e-04,  7.7690e-04,  2.7982e-04,
          7.0491e-04,  5.3742e-04, -3.8210e-04,  5.6359e-04,
        [-2.6656e-04,  6.1130e-04,  2.2434e-04, -2.6311e-05, -
          5.6756e-04,  4.3612e-04, -3.1499e-04,  4.5160e-04,
        [-2.0899e-04,  4.8126e-04,  1.7332e-04, -1.9127e-05, -
          4.4133e-04,  3.4002e-04, -2.4542e-04,  3.5118e-04,
        [-2.2124e-04,  5.1352e-04,  1.8316e-04, -1.8215e-05, -
          4.6535e-04,  3.5562e-04, -2.5256e-04,  3.7205e-04,
        [-2.3950e-04,  5.5784e-04,  2.0116e-04, -1.9356e-05, -
          5.0320e-04,  3.8002e-04, -2.6629e-04,  4.0410e-04,
        [-2.8113e-04,  6.4748e-04,  2.3832e-04, -2.6738e-05, -
          5.9495e-04,  4.5302e-04, -3.2400e-04,  4.7511e-04,
        [-2.2247e-04,  5.1250e-04,  1.8589e-04, -2.0382e-05, -
          4.6567e-04,  3.5762e-04, -2.5898e-04,  3.7052e-04,
        [-2.1190e-04,  4.8698e-04,  1.7599e-04, -2.0002e-05, -
          4.4859e-04,  3.4570e-04, -2.4971e-04,  3.5689e-04,
        [-2.9622e-04,  6.8352e-04,  2.4727e-04, -2.6797e-05, -
          6.2590e-04,  4.7932e-04, -3.4325e-04,  4.9932e-04,
        [-2.7135e-04,  6.2463e-04,  2.2625e-04, -2.5299e-05, -
          5.7555e-04,  4.4196e-04, -3.1743e-04,  4.5866e-04,
       grad_fn=<CatBackward0>)

w_q_true is tensor([[-3.3509e-04,  7.7690e-04,  2.7982e-04, -2
```

```
          7.0490e-04,  5.3742e-04, -3.8210e-04,  5.6359e-04,
        [-2.6656e-04,  6.1130e-04,  2.2434e-04, -2.6311e-05, -
          5.6756e-04,  4.3612e-04, -3.1499e-04,  4.5160e-04,
        [-2.0899e-04,  4.8126e-04,  1.7332e-04, -1.9127e-05, -
          4.4133e-04,  3.4002e-04, -2.4542e-04,  3.5118e-04,
        [-2.2124e-04,  5.1352e-04,  1.8316e-04, -1.8215e-05, -
          4.6535e-04,  3.5562e-04, -2.5256e-04,  3.7206e-04,
        [-2.3950e-04,  5.5784e-04,  2.0116e-04, -1.9356e-05, -
          5.0320e-04,  3.8002e-04, -2.6629e-04,  4.0410e-04,
        [-2.8113e-04,  6.4748e-04,  2.3832e-04, -2.6738e-05, -
          5.9495e-04,  4.5302e-04, -3.2400e-04,  4.7511e-04,
        [-2.2247e-04,  5.1250e-04,  1.8589e-04, -2.0382e-05, -
          4.6567e-04,  3.5762e-04, -2.5898e-04,  3.7052e-04,
        [-2.1190e-04,  4.8698e-04,  1.7599e-04, -2.0002e-05, -
          4.4859e-04,  3.4570e-04, -2.4971e-04,  3.5690e-04,
        [-2.9622e-04,  6.8352e-04,  2.4727e-04, -2.6797e-05, -
          6.2590e-04,  4.7932e-04, -3.4325e-04,  4.9932e-04,
        [-2.7135e-04,  6.2463e-04,  2.2625e-04, -2.5299e-05, -
          5.7555e-04,  4.4196e-04, -3.1743e-04,  4.5866e-04,

W^Q error is tensor([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]], dtype=torch.int16)
```

Top →