Names: Philipp Köhler, Alexander Bespalov

1

a) $f(x) = \left(\frac{x^2}{\log(x)} + c\right) \cdot \left(\frac{x^2}{\log(x)} - c\right)$

$a(x) = \left(\frac{x^2}{\log(x)} + c\right)$

$b(x) = \left(\frac{x^2}{\log(x)} - c\right)$

$c(x) = \frac{x^2}{\log(x)}$

$$\frac{\partial f(x)}{\partial x} = \frac{\partial f(x)}{\partial a(x)} \frac{\partial a(x)}{\partial c(x)} \left(\frac{\partial c(x)}{\partial x^2} \frac{\partial x^2}{\partial x} + \frac{\partial c(x)}{\partial \frac{1}{\ln(x)}} \frac{\partial \frac{1}{\ln(x)}}{\partial \ln(x)} \frac{\partial \ln(x)}{x} \right)$$

$$+ \frac{\partial f(x)}{\partial b(x)} \frac{\partial b(x)}{\partial c(x)} \left(\frac{\partial c(x)}{\partial x^2} \frac{\partial x^2}{\partial x} + \frac{\partial c(x)}{\partial \frac{1}{\ln(x)}} \frac{\partial \frac{1}{\ln(x)}}{\partial \ln(x)} \frac{\partial \ln(x)}{x} \right)$$

$$= \left(\frac{\partial f(x)}{\partial a(x)} \frac{\partial a(x)}{\partial c(x)} + \frac{\partial f(x)}{\partial b(x)} \frac{\partial b(x)}{\partial c(x)} \right) \left(\frac{\partial c(x)}{\partial x^2} \frac{\partial x^2}{\partial x} + \frac{\partial c(x)}{\partial \frac{1}{\ln(x)}} \frac{\partial \frac{1}{\ln(x)}}{\partial \ln(x)} \frac{\partial \ln(x)}{\partial x} \right)$$

b)

$x = 3$, $c = 5$

$x^2 = 9$, $\ln(x) = \ln(3) \approx 1,1$

$c(3) = \frac{9}{\ln(3)} \approx 8,2$

$a(3) = \frac{9}{\ln(3)} + 5 \approx 13,2$, $b(3) = \frac{9}{\ln(3)} - 5 \approx 3,2$

$f(3) \approx 42,1$

c)

$$\left.\frac{\partial \ln(x)}{\partial x}\right|_{x=3} = \left.\frac{1}{x}\right|_{x=3} = \frac{1}{3}$$

$$\left.\frac{\partial \frac{1}{\ln(x)}}{\partial \ln(x)}\right|_{\ln(x) = \ln(3) \approx 1,1} = -\frac{1}{\ln(3)^2} \approx -0,8$$

$$\left.\frac{\partial c(x)}{\partial \frac{1}{\ln(x)}}\right|_{x^2=9} = \left. x^2 \right|_{x^2=9} = 9$$

$$\left.\frac{\partial x^2}{\partial x}\right|_{x=3} = \left. 2x \right|_{x=3} = 6$$

$$\left.\frac{\partial c(x)}{\partial x^2}\right|_{\frac{1}{\ln(x)} = \frac{1}{\ln(3)} \approx 0,9} = \left.\frac{1}{\ln(x)}\right|_{\frac{1}{\ln(x)} = \frac{1}{\ln(3)} \approx 0,9} = 0,9$$

$\frac{\partial b(x)}{\partial c(x)} = 1$, $\frac{\partial a(x)}{\partial c(x)} = 1$

$$\left.\frac{\partial f(x)}{\partial b(x)}\right|_{a(x) = a(3) \approx 13,2} = \left. a(x) \right|_{a(x) = a(3) \approx 13,2} \approx 13,2$$

$$\left.\frac{\partial f(x)}{\partial a(x)}\right|_{b(x)=b(3)\approx 3,2} = b(x)\Big|_{b(x)=b(3)\approx 3,2} \approx 3,2$$

$$\Rightarrow \left.\frac{\partial f(x)}{\partial x}\right|_{x=3,\, c=5} \approx (3,2 \cdot 1 + 13,2 \cdot 1) \cdot \left(0,5 \cdot 6 + 9 \cdot (-0,8) \cdot \frac{1}{3}\right) = 45,2$$

In this case and by hand symbolic differentiation is easier, because a lot of annotation disappears.

d)

1 Reverse Mode Automatic Differentiation

d)

```python
import torch

x = torch.tensor(3.0, requires_grad=True)
c = torch.tensor(5.0, requires_grad=True)

f1 = x**2
f2 = torch.log(x)
f3 = f1 / f2
f4 = f3 + c
f5 = f3 - c
output = f4 * f5

output.backward()

dx = x.grad
dc = c.grad

print(f"Derivative with respect to x: {dx}")
print(f"Derivative with respect to c: {dc}")
```

```
Derivative with respect to x: 48.756893157958984
Derivative with respect to c: -10.0
```

solution is different from calculated x with 0.5 distance, which is due to the rounding of intermediate steps in the calculation by hand.

2

a)

$$w^{t+1} = w^t - \alpha \frac{\hat{m}^t}{\sqrt{\hat{v}^t} + \varepsilon}$$

Update of the parameters with learningrate α.

$$m^t = \beta m^{t-1} + (1-\beta) g^t \quad , \quad \hat{m}^t = \frac{m^t}{(1-(\beta)^t)}$$

$\hat{m}^t$ is the momentum that smooths the gradient g^t with the hyperparameter $\beta \in [0,1]$

$$v^t = \gamma v^{t-1} + (1-\gamma) (g^t)^2 \quad , \quad \hat{v}^t = \frac{v^t}{(1-(\gamma)^t)}$$

$\hat{v}^t$ reduces the gradient for steep gradients

The division of $(1-(\beta)^t)$ and $(1-(\gamma)^t)$ counteracts the initialization bias towards 0.

b)

$m^0 = 0 \quad , \quad v^0 = 0$

$$\hat{m}^1 = (\beta m^0 + (1-\beta) g^1) / (1-(\beta)^1) = g$$

$$\hat{v}^1 = (\gamma v^0 + (1-\gamma) (g^1)^2) / (1-(\gamma)^1) = g^2$$

$$\frac{\hat{m}^1}{\sqrt{\hat{v}^1} + \varepsilon} = \frac{g}{\sqrt{g^2} + \varepsilon} = \frac{g}{|g|} = \text{sign}(g)$$

c)

$m^1 = (1-\beta) g^1 \quad , \quad v^1 = (1-\gamma)(g^1)^2$

$$\hat{m}^2 = (\beta m^1 + (1-\beta) g^2) / (1-(\beta)^2) = \frac{\beta(1-\beta)}{(1-\beta^2)} g^1 + \frac{(1-\beta)}{(1-\beta^2)} g^2$$

$$\hat{v}^2 = (\gamma v^1 + (1-\gamma) (g^2)^2) / (1-(\gamma)^2) = \frac{\gamma(1-\gamma)}{(1-\gamma^2)} (g^1)^2 + \frac{1-\gamma}{(1-\gamma^2)} (g^2)^2$$

$$\frac{\hat{m}^2}{\sqrt{\hat{v}^2} + \varepsilon} = \frac{\frac{\beta(1-\beta)}{(1-\beta^2)} g^1 + \frac{(1-\beta)}{(1-\beta^2)} g^2}{\sqrt{\frac{\gamma(1-\gamma)}{(1-\gamma^2)} (g^1)^2 + \frac{1-\gamma}{(1-\gamma^2)} (g^2)^2} + \varepsilon}$$

d)

Smaller initial learningrates in the initial steps (learningrate warmup) can solve this issue of the dominating sign(g).

e)

Adam has an adaptive learningrate therefore L2 regularization is not the same as weight decay. In the case of SGD it would be. In Adam the regularization is varied by the adaptive learningrate wich leads to inconsistency and less predictable behaviour. The weight decay is preferred.

3

a) maxpooling Kernel $k = 2$, with stride $s = 2$, padding $p = 0$

convolution Kernel $k = 3$, with stride $s = 1$, padding $p = 1$

receptive field r

$r_{out} = r_{in} + (k-1) \cdot jump$, $jump_{out} = jump_{in} \cdot s$, general formula:

$$r_L = 1 + \sum_{i=1}^{L} \prod_{j=1}^{i-1} s_j \cdot (k_i - 1)$$

initial: $r_0 = 1$, $jump_0 = 1$

Conv 1: $r_1 = r_0 + (k-1) \cdot jump_0 = 1 + (3-1) \cdot 1 = 3$, $jump_1 = jump_0 \cdot s = 1 \cdot 1 = 1$

Conv 2: $r_2 = 3 + (3-1) \cdot 1 = 5$, $jump_2 = 1 \cdot 1 = 1$

maxp. 1: $r_3 = 5 + (2-1) \cdot 1 = 6$, $jump_3 = 1 \cdot 2 = 2$

Conv. 3: $r_4 = 6 + (3-1) \cdot 2 = 10$, $jump_4 = 2 \cdot 1 = 2$

Conv. 4: $r_5 = 10 + (3-1) \cdot 2 = 14$, $jump_5 = 2 \cdot 1 = 2$

maxp. 2: $r_6 = 14 + (2-1) \cdot 2 = 16$, $jump_6 = 2 \cdot 2 = 4$

$\vdots$

$r_{out} = 212$

3 Receptive Field of VGG16

a)

```python
arch = [0,0,1,0,0,1,0,0,0,1,0,0,0,1,0,0,0,1]
r = 1
j = 1

for layer in arch:
    if layer == 0: # 0 for conv
        k = 3
        s = 1
    elif layer == 1: # 1 for maxpool
        k = 2
        s = 2
    else:
        print("not known layer")
    r += (k-1)*j
    j *= s

r
```

[1] ✓ 0.0s

```
212
```

b)

Params in conv layer = # Filters · [# input channels · Kernel size + $\underbrace{1}_{\text{Bias}}$]

maxpool no params

Params in FC layer = # inputs · # outputs + # outputs

First layer:

$\# param = 64 \cdot (3 \cdot 3 + 1) = 640$

2. layer: $\# param = 64 \cdot (64 \cdot 3 + 1) = 12\,352$

$\vdots$

total params = 129 066 664

fc params = 123 642 856

conv params = 5 423 808

$$ratio = \frac{\# conv\ param}{\# fc\ params} = 0{,}044$$

b)

```python
def conv_param(filters, inp_chan, k):
    return filters * (inp_chan * k +1)

def fc_param(inputs, outputs):
    return inputs * outputs + outputs

layers_conv = [[64,3,3],[64,64,3],[128,128,3],
               [128,128,3],[256,256,3],[256,256,3],
               [256,256,3],[512,512,3],[512,512,3],
               [512,512,3],[512,512,3],[512,512,3],
               [512,512,3]]

conv_params = 0
for layer in layers_conv:
    conv_params += conv_param(*layer)

layers_fc = [[7*7*512,4096],[4096,4096],[4096,1000]]

fc_params = 0
for layer in layers_fc:
    fc_params += fc_param(*layer)

print(f"# total params: {fc_params+conv_params}")
print(f"# fc params: {fc_params}")
print(f"# conv params: {conv_params}")
print(f"ratio: {conv_params/fc_params}")
```

[2] ✓ 0.0s

```
# total params: 129066664
# fc params: 123642856
# conv params: 5423808
ratio: 0.043866731774620284
```