Homework I for the Deep Learning Class

Anastasiia Kasprova

1. Write down the neural network's forward pass in scalar form using (1-5) equations. Show it as an evolution of the input vector that passes layer-by-layer through the entire network to the output layer, showing the size of each vector or matrix. Indices in summation operators must also be specified.

Input tensor in scalar form:

$$x = \sum_{n=1}^{N_{batch}} \sum_{c_{in}=1}^{C_{in}} \sum_{m=1}^{S_{in}} \sum_{l=1}^{S_{in}} x_{n,c_{in},m,l} = \sum_{n=1}^{64} \sum_{c_{in}=1}^{1} \sum_{m=1}^{28} \sum_{l=1}^{28} x_{n,c_{in},m,l} \tag{1}$$

$$dim(X) : [N_{batch} \times C_{in} \times S_{in} \times S_{in}] = [64 \times 1 \times 28 \times 28] \tag{2}$$

N_{batch} - batch size
C_{in} - number of input channels
S_{in} - input image size

1.1 Convolutional Layer (conv_layer)
Parameters:
number of filters = 20, $kernel\_size = 5$, $stride = 1$, $padding = 0$, C_{out}=B
Equation of convolution:

$$z^{(conv)}_{n,c_{in},m,l} = \sum_{c_{in}=1}^{C_{in}} \sum_{i=1}^{K} \sum_{j=1}^{K} x_{n,c_{in},m+i-1,l+j-1} w^{(conv)}_{c_{out},c_{in},i,j} + b^{(conv)}_{c_{out}} \tag{3}$$

Weight tensor

$$w^{(conv)} = \sum_{c_{out}=1}^{C_{out}} \sum_{c_{in}=1}^{C_{in}} \sum_{i=1}^{K} \sum_{j=1}^{K} w^{(conv)}_{c_{out},c_{in},i,j} \tag{4}$$

$$dim(w^{(conv)}) : [C_{out} \times C_{in} \times K \times K] = [B \times C_{in} \times K \times K] = [20 \times 1 \times 5 \times 5] \tag{5}$$

C_{out} - number of output channels
C_{in} - number of input channels
K - kernel size
Bias vector

$$b^{(conv)} = \sum_{c_{out}=1}^{C_{out}} b^{(conv)}_{c_{out}} \tag{6}$$

$$dim(b^{(conv)}) : [C_{out}] = [B] = [20] \tag{7}$$

Output tensor of convolution

$$z^{(conv)} = \sum_{n=1}^{N_{batch}} \sum_{c_{in}=1}^{C_{in}} \sum_{m=1}^{S_{out}} \sum_{l=1}^{S_{out}} z^{(conv)}_{n,c_{in},m,l} \tag{8}$$

$$dim(z^{(conv)}) : [N_{batch} \times C_{out} \times S_{out} \times S_{out}] =$$

$= [\mathrm{N}_{batch} \times B \times \frac{S_{in}-K+2P}{S} + 1 \times \frac{S_{in}-K+2P}{S} + 1] =$
$= [64 \times 20 \times \frac{28-5+0}{1} + 1 \times \frac{28-5+0}{1} + 1] =$
$= [64 \times 20 \times 24 \times 24]$(9)
S_{in} - size of input image
P - padding
K - kernel size
S - stride

1.2 Max Pooling Layer (max_pool_2d_layer)

Parameters:
$kernel\_size = 2$, $stride = 2$, $padding = 0$
Equation of max-pooling:

$$z^{(pool)}_{n,c_{in},m,l} = max(z^{(conv)}_{n,c_{out},2i-1,2j-1}, z^{(conv)}_{n,c_{out},2i-1,2j}, z^{(conv)}_{n,c_{out},2i,2j-1}, z^{(conv)}_{n,c_{out},2i,2j}) \tag{10}$$

$$z^{(pool)} = \sum_{n=1}^{N_{batch}} \sum_{c_{in}=1}^{C_{in}} \sum_{m=1}^{S_{out}} \sum_{l=1}^{S_{out}} z^{(pool)}_{n,c_{in},m,l} = \sum_{n=1}^{64} \sum_{c_{in}=1}^{20} \sum_{m=1}^{12} \sum_{l=1}^{12} z^{(pool)}_{n,c_{in},m,l} \tag{11}$$

$$dim(z^{(pool)}) : [N_{batch} \times C_{in} \times S_{out} \times S_{out}] =$$

$= [\mathrm{N}_{batch} \times B \times \frac{S_{in}-K+2P}{S} + 1 \times \frac{S_{in}-K+2P}{S} + 1] =$
$= [64 \times 20 \times \frac{24-2+0}{2} + 1 \times \frac{24-2+0}{2} + 1] =$
$= [64 \times 20 \times 12 \times 12]$(12)

1.3 Reshape Layer (reshape_layer)

Equation of reshape:

$$z^{(reshaped)}_{n,j} = z^{(pool)}_{n,c_{in},m,l} \tag{13}$$

$$j = (c_{in} - 1) * S_{in} * S_{in} + (m-1) * S_{in} + l =$$

$= \mathrm{c}_{in} * S_{in} * S_{in} - S_{in} * S_{in} + S_{in} * S_{in} - S_{in} + S_{in} =$
$= \mathrm{c}_{in} * S_{in} * S_{in} =$
$= 20 * 12 * 12 = 2880$(14)

$$z^{(reshaped)} = \sum_{n=1}^{N_{batch}} \sum_{j=1}^{C_{in}*S_{in}*S_{in}} z^{(reshaped)}_{n,j} = \sum_{n=1}^{64} \sum_{j=1}^{2880} z^{(reshaped)}_{n,j} \tag{15}$$

$$dim(z^{(reshaped)}) : [N_{batch} \times C_{in} * S_{in} * S_{in}] = [64 \times 2880] \tag{16}$$

1.4 Fully connected Layer 1 (fc1_layer)
Equation of fully-connection:

$$z_{n,j}^{(fc1)} = \sum_{i=1}^{D} w_{j,i}^{(fc1)} * z_{n,i}^{(reshaped)} + b_j^{(fc1)} \tag{17}$$

Parameters:
number of output units = 500
FC 1 Weight matrix

$$w^{(fc1)} = \sum_{j=1}^{P} \sum_{i=1}^{D} w_{j,i}^{(fc1)} = \sum_{j=1}^{500} \sum_{i=1}^{2880} w_{j,i}^{(fc1)} \tag{18}$$

$$dim(w^{(fc1)}) : [P \times D] = [500 \times 2880] \tag{19}$$

D - number of inputs
P - number of outputs
FC 1 Bias

$$b^{(fc1)} = \sum_{j=1}^{P} b_j^{(fc1)} = \sum_{j=1}^{500} b_j^{(fc1)} \tag{20}$$

$$dim(b^{(fc1)}) : [P] = [500] \tag{21}$$

FC 1 layer's output

$$z^{(fc1)} = \sum_{n=1}^{N_{batch}} \sum_{j=1}^{P} z_{n,j}^{(fc1)} = \sum_{n=1}^{64} \sum_{j=1}^{500} z_{n,j}^{(fc1)} \tag{22}$$

$$dim(z^{(fc1)}) : [N_{batch} \times P] = [64 \times 500] \tag{23}$$

1.5 Rectified linear units (ReLU) Layer (relu_layer)
The size of vector does not change.

$$z_{n,j}^{(relu)} = relu(z_{n,j}^{(fc1)}) \tag{24}$$

$$z^{(relu)} = \sum_{n=1}^{N_{batch}} \sum_{j=1}^{D} z_{n,j}^{(relu)} \tag{25}$$

$$dim(z^{(relu)}) : [N_{batch} \times D] = [64 \times 500] \tag{26}$$

D - number of inputs

1.6 Fully connected Layer 2 (fc2_layer)
Equation of fully-connection:

$$z_{n,j}^{(fc2)} = \sum_{i=1}^{D} w_{j,i}^{(fc2)} * z_{n,i}^{(relu)} + b_j^{(fc2)} \tag{27}$$

Parameters:
number of output units = 10
FC 2 Weight matrix

$$w^{(fc2)} = \sum_{j=1}^{P} \sum_{i=1}^{D} w_{j,i}^{(fc2)} = \sum_{j=1}^{10} \sum_{i=1}^{500} w_{j,i}^{(fc2)} \tag{28}$$

$$dim(w^{(fc2)}) : [P \times D] = [10 \times 500] \tag{29}$$

D - number of inputs
P - number of outputs
FC 2 Bias

$$b^{(fc2)} = \sum_{j=1}^{P} b_j^{(fc2)} = \sum_{j=1}^{10} b_j^{(fc2)} \tag{30}$$

$$dim(b^{(fc2)}) : [P] = [10] \tag{31}$$

FC 2 layer's output

$$z^{(fc2)} = \sum_{n=1}^{N_{batch}} \sum_{j=1}^{P} z_{n,j}^{(fc1)} = \sum_{n=1}^{64} \sum_{j=1}^{10} z_{n,j}^{(fc2)} \tag{32}$$

$$dim(z^{(fc2)}) : [N_{batch} \times P] = [64 \times 10] \tag{33}$$

1.7 Softmax Layer (softmax_layer)
The size of vector does not change.

$$y_{n,j} = softmax(z_{n,j}^{(fc2)}) \tag{34}$$

$$y = \sum_{n=1}^{N_{batch}} \sum_{j=1}^{D} z_{n,j}^{(fc2)} \tag{35}$$

$$dim(y) : [N_{batch} \times D] = [64 \times 10] \tag{36}$$

D - number of inputs

2. Write down the neural network's forward pass in vector form:

(a) for the convolutional layer described in section 2.1 on "moving window" level. Replace "moving window" algorithm by matrix multiplication. Please, use 'im2col' trick. Column's length in reshaped input matrix is $K * K$

(b) for the pooling layer described in 2.2

(c) for fully connected layer described in 2.4

(d*) write down convolution as matrix multiplication adding parallelization on channel level (column's length in reshaped input matrix is C_{in} *K*K)

2.a Fully-connected layer with 'im2col' trick

Vector form of convolutional layer with applied 'im2col' trick on "moving window" level (NB! **W** and $\mathbf{X}_{\mathbf{conv}}$ reshaped according to 'im2col' requirements):

$$\mathbf{Z}^{(conv)} = \mathbf{W}\mathbf{X}^{(conv)} + \mathbf{B}^{(conv)} \tag{37}$$

$dim(\mathbf{X}) : [K * K \times (\frac{S_{in}-K+2P}{S} + 1) * (\frac{S_{in}-K+2P}{S} + 1)] = [5 * 5 \times (\frac{28-5+0}{1} + 1) * (\frac{28-5+0}{1} + 1)] = [25 \times 24 * 24] = [25 \times 576]$

$dim(\mathbf{W}^{(\mathbf{conv})}) : [1 \times K * K] = [1 \times 5 * 5] = [1 \times 25]$

$dim(\mathbf{B}^{(\mathbf{conv})}) : [1 \times S_{out} * S_{out}] =$
$= [1 \times (\frac{S_{in}-K+2P}{S} + 1) * (\frac{S_{in}-K+2P}{S} + 1)] =$
$= [1 \times (\frac{28-5+0}{1} + 1) * (\frac{28-5+0}{1} + 1)] =$
$= [1 \times 24 * 24] = [1 \times 576]$

$dim(\mathbf{Z}^{(\mathbf{conv})}) : [1 \times S_{out} * S_{out}] =$
$= [1 \times (\frac{S_{in}-K+2P}{S} + 1) * (\frac{S_{in}-K+2P}{S} + 1)] =$
$= [1 \times (\frac{28-5+0}{1} + 1) * (\frac{28-5+0}{1} + 1)] =$
$= [1 \times 24 * 24] = [1 \times 576]$

Shown above dimensions of matricies are applicable for a single image input. Since we deal with N_{batch}:

$dim(\mathbf{Z}^{(\mathbf{conv})}) = [64 \times 576]$
$dim(\mathbf{B}^{(\mathbf{conv})}) = [64 \times 576]$
$dim(\mathbf{W}^{(\mathbf{conv})}) = [64 \times 25]$

2.b Pooling layer in vector form

General equation with pooling in vector form

$$\mathbf{Z}^{(1)} = f(\mathbf{A}^{(1)}) \tag{38}$$

In our case:

$$\mathbf{Z}^{(pool)} = f(\mathbf{Z}^{(conv)}) \tag{39}$$

Dimensionalities explained in (1.2)

2.c Fully-connected layer in vector form

$$\mathbf{Z}^{(fc1)} = \mathbf{X}^{(reshaped)}\mathbf{W}^{-T} + \mathbf{B}^{(fc1)} \tag{40}$$

$dim(\mathbf{Z^{(fc1)}}) = [64 \times 500]$
$dim(\mathbf{X^{(reshaped)}}) = [64 \times 2880]$
$dim(\mathbf{W^T}) = [2880 \times 500]$
$dim(\mathbf{B^{(fc1)}}) = [64 \times 500]$

2.d* Convolution as matrix multiplication with parallelization on channel level

$$\mathbf{Z}^{(conv)} = \mathbf{W}\mathbf{X}^{(conv)} + \mathbf{B}^{(conv)} \tag{41}$$

$dim(\mathbf{X}) : [C_{in} * K * K \times (\frac{S_{in}-K+2P}{S} + 1) * (\frac{S_{in}-K+2P}{S} + 1)] = [1 * 5 * 5 \times (\frac{28-5+0}{1} + 1) * (\frac{28-5+0}{1} + 1)] = [25 \times 24 * 24] = [25 \times 576]$
$dim(\mathbf{W^{(conv)}}) : [1 \times C_{in} * K * K] = [1 \times 1 * 5 * 5] = [1 \times 25]$
$dim(\mathbf{B^{(conv)}}) : [1 \times S_{out} * S_{out}] = [1 \times (\frac{S_{in}-K+2P}{S} + 1) * (\frac{S_{in}-K+2P}{S} + 1)] = [1 \times (\frac{28-5+0}{1} + 1) * (\frac{28-5+0}{1} + 1)] = [1 \times 24 * 24] = [1 \times 576]$
$dim(\mathbf{Z^{(conv)}}) : [1 \times S_{out} * S_{out}] = [1 \times (\frac{S_{in}-K+2P}{S} + 1) * (\frac{S_{in}-K+2P}{S} + 1)] = [1 \times (\frac{28-5+0}{1} + 1) * (\frac{28-5+0}{1} + 1)] = [1 \times 24 * 24] = [1 \times 576]$

3. Write down the backward pass in scalar form. Derive the gradients $\frac{\partial Loss}{\partial w^{(conv)}_{c_{out},c_{in},m,l}}$ and $\frac{\partial Loss}{\partial b^{(conv)}_{c_{out}}}$ in (1). No need to differentiate the forward pass equations, use the "delta-rule" instead. Local gradient for the output softmax layer is:

$$\delta_k^{(out)} \equiv \frac{\partial Loss}{\partial z_k^{(fc2)}} = y_k - t_k \tag{42}$$

where y_k is k-th model's output, t_k is k-th target value. Local gradients *partial* for all intermediate layers must be shown.

Layer by layer differentiation

Considering Softmax (34), FC2 (27) layers and MSE function general representation, find the derivatives of error function with respect to weights of the fc2 layer:

$$z_{n,j}^{(fc2)} = \sum_{s=1}^{S} w_{j,s}^{(fc2)} z_{n,s}^{(relu)} + b_j^{(fc2)} \tag{43}$$

$$y_{n,j} = s(z_{n,j}^{(fc2)}) \tag{44}$$

$$E(w) = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{L}(y(x_{n,k}, w) - t_{n,k})^2 \tag{45}$$

n - number of training sample in dataset, k - number of network's output

$$\frac{\partial E}{\partial w_{k,s}^{(fc2)}} = (y_k - t_k)\frac{\partial y_k}{\partial w_{k,s}^{(fc2)}} \tag{46}$$

Applying the chain rule to $\frac{\partial y_k}{\partial w_{k,s}^{(fc2)}}$:

$$\frac{\partial E}{\partial w_{k,s}^{(fc2)}} = (y_k - t_k)\frac{\partial s(z_k^{(fc2)})}{\partial z_k^{(fc2)}}\frac{\partial z_k^{(fc2)}}{\partial w_{k,s}^{(fc2)}} \tag{47}$$

$$\frac{\partial E}{\partial w_{k,s}^{(fc2)}} = (y_k - t_k)s'(z_k^{(fc2)})z_s^{(relu)} \tag{48}$$

Introducing deltas to fc2 layer:

$$\delta_k^{(fc2)} = (y_k - t_k)s'(z_k^{(fc2)}) \tag{49}$$

where s - is a softmax function
Thus, we have:

$$\frac{\partial E}{\partial w_{k,s}^{(fc2)}} = \delta_k^{(fc2)} z_s^{(relu)} \tag{50}$$

Considering fc1 (17) and relu (24) layers find the derivatives of the error function with respect to weights of the fc1 layer:

$$\frac{\partial E}{\partial w_{s,r}^{(fc1)}} = \sum_{k=1}^{L}(y_k - t_k)\frac{\partial y_k}{\partial w_{s,r}^{(fc1)}} \tag{51}$$

Applying the chain rule:

$$\frac{\partial E}{\partial w_{s,r}^{(fc1)}} = \sum_{k=1}^{L}(y_k - t_k)\frac{\partial s(z_k^{(fc2)})}{\partial z_k^{(fc2)}}\frac{\partial z_k^{(fc2)}}{\partial w_{s,r}^{(fc1)}} \tag{52}$$

$$\frac{\partial E}{\partial w_{s,r}^{(fc1)}} = \sum_{k=1}^{L}(y_k - t_k)\frac{\partial s(z_k^{(fc2)})}{\partial z_k^{(fc2)}}\frac{\partial z_k^{(fc2)}}{\partial z_s^{(relu)}}\frac{\partial z_s^{(relu)}}{\partial w_{s,r}^{(fc1)}} \tag{53}$$

$$\frac{\partial E}{\partial w_{s,r}^{(fc1)}} = \sum_{k=1}^{L}(y_k - t_k)\frac{\partial s(z_k^{(fc2)})}{\partial z_k^{(fc2)}}\frac{\partial z_k^{(fc2)}}{\partial z_s^{(relu)}}\frac{\partial r(z_s^{(fc1)})}{\partial z_s^{(fc1)}}\frac{\partial z_s^{(fc1)}}{\partial w_{s,r}^{(fc1)}} \tag{54}$$

$$\frac{\partial E}{\partial w_{s,r}^{(fc1)}} = \sum_{k=1}^{L}(y_k - t_k)\frac{\partial s(z_k^{(fc2)})}{\partial z_k^{(fc2)}}\frac{\partial z_k^{(fc2)}}{\partial z_s^{(relu)}}\frac{\partial r(z_s^{(fc1)})}{\partial z_s^{(fc1)}} z_r^{(reshaped)} \tag{55}$$

$$\frac{\partial E}{\partial w_{s,r}^{(fc1)}} = \sum_{k=1}^{L}(y_k - t_k)s'(z_k^{(fc2)})w_{k,s}r'(z_s^{(fc1)})z_r^{(reshaped)} \tag{56}$$

Introducing deltas to fc1 layer:

$$\delta_s^{(fc1)} = \sum_{k=1}^{L}(y_k - t_k)s'(z_k^{(fc2)})w_{k,s}r'(z_s^{(fc1)}) \tag{57}$$

$$\delta_s^{(fc1)} = relu'(z_r^{(fc1)})\sum_{k=1}^{20}(y_k - t_k)w_{k,s}softmax'(z_k^{(fc2)}) \tag{58}$$

$$\frac{\partial E}{\partial w_{s,r}^{(fc1)}} = \delta_s^{(fc1)} z_r^{(reshaped)} \tag{59}$$

Considering convolutional (3) and max pooling (10) layers find the derivatives of the error function with respect to weights of the convolutional layer:

$$\frac{\partial E}{\partial w_{r,p}^{(conv)}} = \sum_{s=1}^{D}\delta_s^{(fc1)}\frac{\partial z_s^{(fc1)}}{\partial w_{r,p}^{(conv)}} \tag{60}$$

$$\frac{\partial E}{\partial w_{r,p}^{(conv)}} = \sum_{s=1}^{D}\delta_s^{(fc1)}\frac{\partial z_s^{(fc1)}}{\partial z_r^{(reshaped)}}\frac{\partial z_r^{(reshaped)}}{\partial w_{r,p}^{(conv)}} \tag{61}$$

$$\frac{\partial E}{\partial w_{r,p}^{(conv)}} = \sum_{s=1}^{D}\delta_s^{(fc1)}w_{s,r}^{(fc1)}\frac{\partial z_r^{(reshaped)}}{\partial w_{r,p}^{(conv)}} \tag{62}$$

Considering reshape (13), which does not change the activations, but only change tensors dimensions to be compatible with other layers :

$$z_{r,p}^{(reshaped)} = z_{n,c_{in},m,l}^{(pool)} \tag{63}$$

$$\frac{\partial E}{\partial w_{c_{out},c_{in},m,l}^{(conv)}} = \sum_{s=1}^{D}\delta_s^{(fc1)}w_{s,r}^{(fc1)}\frac{\partial z_{n,c_{in},m,l}^{(pool)}}{\partial w_{c_{out},c_{in},m,l}^{(conv)}} \tag{64}$$

$$\frac{\partial E}{\partial w_{c_{out},c_{in},m,l}^{(conv)}} = \sum_{s=1}^{D}\delta_s^{(fc1)}w_{s,r}^{(fc1)}p'(z_{c_{out},c_{in},m,l}^{(conv)})\frac{\partial z_{n,c_{in},m,l}^{(conv)}}{\partial w_{c_{out},c_{in},m,l}^{(conv)}} \tag{65}$$

$$\frac{\partial E}{\partial w^{(conv)}_{c_{out},c_{in},m,l}} = \sum_{s=1}^{D} \delta_s^{(fc1)} w_{s,r}^{(fc1)} p'(z^{(conv)}_{c_{out},c_{in},m,l}) \sum_{i,j=1}^{K} x_{n,c_{in},m+i-1,l+j-1} \quad (66)$$

Introducing deltas to convolutional layer:

$$\delta^{(conv)}_{c_{out},c_{in},m,l} = \sum_{s=1}^{D} \delta_s^{(fc1)} w_{s,r}^{(fc1)} p'(z^{(conv)}_{c_{out},c_{in},m,l}) \quad (67)$$

$$\delta^{(conv)}_{c_{out},c_{in},m,l} = p'(z^{(conv)}_{c_{out},c_{in},m,l}) \sum_{s=1}^{D} \delta_s^{(fc1)} w_{s,r}^{(fc1)} = maxpool'(z^{(conv)}_{c_{out},c_{in},m,l}) \sum_{s=1}^{500} \delta_s^{(fc1)} w_{s,r}^{(fc1)} \quad (68)$$

Let us find $\frac{\partial Loss}{\partial b^{(conv)}_{c_{out}}}$ starting from the output layer and applying chain rule:

$$\frac{\partial E}{\partial b_k^{(fc2)}} = (y_k - t_k) s'(z_k^{(fc2)}) \quad (69)$$

As far as we can notice $\frac{\partial E}{\partial b_k^{(fc2)}} = \delta_k^{(fc2)}$.

$$\frac{\partial E}{\partial b_s^{(fc1)}} = r'(z_s^{(fc1)}) \sum_{k=1}^{L} (y_k - t_k) s'(z_k^{(fc2)}) w_{k,s} \quad (70)$$

The same is true for $\frac{\partial E}{\partial b_s^{(fc1)}} = \delta_s^{(fc1)}$.

$$\frac{\partial E}{\partial b^{(conv)}_{c_{out}}} = p'(z^{(conv)}_{c_{out},c_{in},m,l}) \sum_{s=1}^{D} \delta_s^{(fc1)} w_{s,r}^{(fc1)} \quad (71)$$

Thus: $\frac{\partial E}{\partial b^{(conv)}_{C_{out}}} = \delta^{(conv)}_{c_{out},c_{in},m,l}$