Introduction to Machine Learning
Problems: Convolutional Neural Networks

Prof. Sundeep Rangan

1. *Tensors.* For each of the following datasets, describe how you would represent them as tensors. Specifically, give the shape of the tensors.

   (a) A batch of 100 color images, each image is 256 $\times$ 256.

   (b) A batch of 40 EEG recordings. Each EEG records has 80 channels of output at a sample rate of 240 Hz for 10 seconds.

   (c) A batch of 32 videos. Each video has a frame rate of 30 frames per second and is 10 seconds long. The video is color with a resolution of 512 $\times$ 512.

   Solution:

   (a) We represent this as `(sample,row,col,color)` for a tensor shape of $(100, 256, 256, 3)$.

   (b) There are $(240)(10) = 2400$ time samples in each recording. So, we represent this as `(sample,time,chan)` for a tensor shape of $(40, 2400, 80)$.

   (c) There are $(30)(10) = 300$ frames in each video. So, we represent this as `(sample,frame,row,col,color)` for a tensor shape of $(32, 300, 512, 512, 3)$.

2. *2D convolutions.* Let X and W be arrays,

$$X = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 3 & 3 & 0 \\ 0 & 3 & 3 & 3 & 0 \\ 0 & 3 & 2 & 3 & 0 \\ 0 & 3 & 2 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad W = \begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix}.$$

   Let Z be the 2D convolution (without reversal):

$$Z[i,j] = \sum_{k_1,k_2} W[k_1,k_2] X[i+k_1, j+k_2]. \tag{1}$$

   Assume that the arrays are indexed starting at $(0,0)$.

   (a) What are the limits of the summations over k_1 and k_2 in (1)?

   (b) What is the size of the output $Z[i,j]$ if the convolution is computed only on the *valid* pixels (i.e. the pixel locations (i,j) where the summation in (1) does not exceed the boundaries of W or X).

(c) What is the largest positive value of $Z[i,j]$ and state one pixel location (i,j) where that value occurs.

(d) What is the largest negative value of $Z[i,j]$ and state one pixel location (i,j) where that value occurs.

(e) Find one pixel location where $Z[i,j] = 0$.

Solution:

(a) Both indices go over the range of $W[k_1, k_2]$: $0 \leq k_1, k_2 < 2$.

(b) Since X is 6×5 and W is 2×2 and we are selecting valid locations only, the size will of Z will be

$$(6-2+1) \times (5-2+1) = 5 \times 4.$$

(c) We have that

$$Z[i,j] = X[i,j] + X[i+1,j] - X[i,j+1] - X[i+1,j+1].$$

So, $Z[i,j]$ will be the largest positive value when there is a large negative change across one column. This occurs at $(i,j) = (1,3)$:

$$Z[1,3] = X[1,3] + X[2,3] - X[1,4] - X[2,4] = 3+3-0-0 = 6.$$

We get the same value at $(2,3)$ and $(3,3)$.

(d) For a negative value, we need there to be a large positive change across one column, which occurs at

$$Z[1,0] = X[1,0] + X[2,0] - X[1,0] - X[2,0] = 0+0-3-3 = -6.$$

We get the same value at $(2,0)$ and $(3,0)$.

(e) You can take $(i,j) = (1,1)$ or $(1,2)$. For example,

$$Z[1,1] = X[1,1] + X[2,1] - X[1,2] - X[2,2] = 3+3-3-3 = 0.$$

3. *Complexity and number of parameters.* Suppose that a convolutional layer of a neural network has an input tensor $X[i,j,k]$ and computes an output via a convolution and ReLU activation,

$$\begin{aligned} Z[i,j,m] &= \sum_{k_1}\sum_{k_2}\sum_{n} W[k_1,k_2,n,m]X[i+k_1, j+k_2, n] + b[m], \\ U[i,j,m] &= \max\{0, Z[i,j,m]\}. \end{aligned}$$

for some weight kernel $W[k_1,k_2,n,m]$ and bias $b[m]$. Suppose that X has shape (48,64,10) and W has shape (3,3,10,20). Assume the convolution is computed on the *valid* pixels.

(a) What are the shapes of Z and U?

(b) What are the number of input channels and output channels?

(c) How many multiplications must be performed to compute the convolution in that layer?

(d) If W and b are to be learned, what are the total number of trainable parameters in the layer?

Solution:

(a) Since each kernel in W is 3×3, each channel of the output is

$$(48 - 3 + 1) \times (64 - 3 + 1) = 46 \times 62.$$

There are 20 output channels, so Z is $46 \times 62 \times 20$.

(b) Since W is $3 \times 3 \times 10 \times 20$, there are 10 input channels and 20 output channels.

(c) Each output of $Z[i, j, m]$ requires summations over the indices

$$0 \le k_1, k_2 < 3, \quad 0 \le n < 10.$$

Therefore, there are (3)(3)(10) multiplications for each output of Z. Since there are (46)(62)(20) outputs, there are a total of

$$(46)(62)(20)(3)(3)(10) = 5.133(10)^6 \text{ multiplications.}$$

You can see why computing outputs in deep networks takes many operations.

(d) The number of parameters in W and b are:

$$\begin{aligned} W: &\quad (3)(3)(10)(20) = 1800 \text{ parameters} \\ b: &\quad 20 \text{ parameters.} \end{aligned}$$

So, there are a total of 1820 parameters.

4. *Back-propagation.* Suppose that a convolutional layer in some neural network is described as a linear convolution followed by a sigmoid activation,

$$\begin{aligned} Z[i, j_1, j_2, m] &= \sum_{k_1} \sum_{k_2} \sum_{n} W[k_1, k_2, n, m] X[i, j_1 + k_1, j_2 + k_2, n] + b[m], \\ U[i, j_1, j_2, m] &= 1/(1 + \exp(-Z[i, j_1, j_2, m])). \end{aligned}$$

where $X[i, j_1, j_2, n]$ is the input of the layer and $U[i, j_1, j_2, m]$ is the output. Suppose that during back-propagation, we have computed the gradient $\partial J / \partial U$ for some loss function J. That is, we have computed the components $\partial J / \partial U[i, j_1, j_2, m]$. Show how to compute the following:

(a) The gradient components $\partial J / \partial Z[i, j_1, j_2, m]$.

(b) The gradient components $\partial J / \partial W[k_1, k_2, n, m]$.

(c) The gradient components $\partial J / \partial X[i, j_1, j_2, n]$.

Solution:

(a) We have

$$\frac{\partial U[i,j_1,j_2,m]}{\partial Z[i,j_1,j_2,m]} = \frac{\exp(-Z[i,j_1,j_2,m])}{(1+\exp(-Z[i,j_1,j_2,m])^2} = U[i,j_1,j_2,m](1-U[i,j_1,j_2,m]).$$

By chain rule,

$$\begin{aligned}\frac{\partial J}{\partial Z[i,j_1,j_2,m]} &= \frac{\partial J}{\partial U[i,j_1,j_2,m]}\frac{\partial U[i,j_1,j_2,m]}{\partial Z[i,j,m]} \\ &= \frac{\partial J}{\partial U[i,j_1,j_2,m]}U[i,j_1,j_2,m](1-U[i,j_1,j_2,m]).\end{aligned}$$

(b) The gradient components $\partial J/\partial W[k_1,k_2,n,m]$. From the convolution equation,

$$\frac{\partial Z[i,j_1,j_2,m]}{\partial W[k_1,k_2,n,m]} = X[i,j_1+k_1,j_2+k_2,n].$$

By chain rule,

$$\begin{aligned}\frac{\partial J}{\partial W[k_1,k_2,n,m]} &= \sum_{j_1,j_2}\frac{\partial J}{\partial Z[i,j_1,j_2,m]}\frac{\partial Z[i,j_1,j_2,m]}{\partial W[k_1,k_2,n,m]} \\ &= \sum_{j_1,j_2}\frac{\partial J}{\partial Z[i,j_1,j_2,m]}X[i,j_1+k_1,j_2+k_2,n].\end{aligned}$$

(c) We want to first compute the partial derivatives,

$$\frac{\partial Z[i,j_1',j_2',m]}{\partial X[i,j_1,j_2,n]},$$

for all output components $Z[i,j_1',j_2',m]$ and inputs $X[i,j_1,j_2,n]$. Note that we had to add the indices j_1',j_2' at the output, to differentiate between the input indices j_1,j_2. To compute this derivative, we need to write $Z[i,j_1',j_2',m]$ in terms of the inputs $X[i,j_1,j_2,n]$. This is matter of re-indexing. First, rewrite the summation in the convolution as,

$$Z[i,j_1',j_2',m] = \sum_{k_1}\sum_{k_2}\sum_{n} W[k_1,k_2,n,m]X[i,j_1'+k_1,j_2'+k_2,n] + b[m].$$

All we have done here is replace j_1,j_2 with j_1',j_2'. Next make the substitution,

$$j_1 = j_1'+k_1, \quad j_2 = j_2'+k_2 \Rightarrow k_1 = j_1-j_1', \quad k_2 = j_2-j_2'.$$

Then, we can sum over j_1,j_2 instead of over k_1,k_2:

$$Z[i,j_1',j_2',m] = \sum_{i}\sum_{j}\sum_{n} W[j_1-j_1',j_2-j_2',n,m]X[i,j_1,j_2,n] + b[m].$$

Now, we have $Z[i, j_1', j_2', m]$ in terms of the inputs $X[i, j_1, j_2, n]$. From this, we see that

$$\frac{\partial Z[i, j_1', j_2', m]}{\partial X[i, j_1, j_2, n]} = W[j_1 - j_1', j_2 - j_2', n, m].$$

Hence, by chain rule,

$$\begin{aligned}\frac{\partial J}{\partial X[i, j_1, j_2, n]} &= \sum_{j_1'} \sum_{j_2'} \sum_{m} \frac{\partial J}{\partial Z[i, j_1', j_2', m]} \frac{\partial Z[i, j_1', j_2', m]}{\partial X[i, j_1, j_2, n]} \\ &= \sum_{j_1'} \sum_{j_2'} \sum_{m} \frac{\partial J}{\partial Z[i, j_1', j_2', m]} W[j_1 - j_1', j_2 - j_2', n, m].\end{aligned}$$

If you got this far, you will get full marks. But, if we let $k_1 = j_1 - j_1'$ and $k_2 = j_2 - j_2'$ and sum over k_1, k_2 instead of j_1', j_2', we get

$$\frac{\partial J}{\partial X[i, j_1, j_2, n]} = \sum_{k_1} \sum_{k_2} \sum_{n} \frac{\partial J}{\partial Z[i, j_1 - k_1, j_2 - k_2, m]} W[k_1, k_2, n, m].$$

We see that the gradient is also a convolution, but with the reversal.

5. *Sub-sampling and pooling.* In CNNs, convolution operations are often followed by a data reduction step, typically either via *sub-sampling* or *max pooling*. The methods can be described as follows: Let $x[j]$, $j = 0, 1, \ldots, N-1$ be a 1D input (say in one channel in one sample). The outputs $y[k]$ for sub-sampling and max-pooling are given by:

   - *Sub-sampling* with *stride* s selects every s-th sample:

   $$y[k] = x[sk], \quad k = 0, 1, \ldots, \left\lfloor \frac{N-1}{s} \right\rfloor.$$

   - *Max pooling* with *pool size* p and *stride* s computes,

   $$y[k] = \max_{j=0,1,\ldots,p-1} x[sk + j], \quad k = 0, 1, \ldots, \left\lfloor \frac{N-1}{s} \right\rfloor.$$

   (a) Let $\mathbf{x}$ be the vector,

   $$\mathbf{x} = [1, 2, 3, 2, 0, 10, 1, 0].$$

   Find the output $\mathbf{y}$ when sub-sampling with stride $s = 2$.

   (b) For the same vector $\mathbf{x}$ as in part (a), find the output of max pooling with stride $s = 2$ and pool size $p = 2$.

   (c) Let $X[i, j, n]$ be a tensor of shape (B, N, C) where B is the batch size, N is the number of samples per channel and C is the number of channels. Write equations for sampling and max pooling of X if the operations are to be performed on each channel and sample. What are the output shapes?

Solution:

(a) Striding selects every second sample: $\mathbf{y} = [1, 3, 0, 1]$. Note this misses the large peak value of 10.

(b) Max pooling selects the maximum of every second sample: $\mathbf{y} = [2, 3, 10, 1]$.

(c) Striding is:

$$Y[i, k, n] = x[i, sk, n], \quad k = 0, 1, \ldots, \left\lfloor \frac{N-1}{s} \right\rfloor,$$

and max pooling is

$$Y[i, k, n] = \max_{j=0,1,\ldots,p-1} X[i, sk + j, n], \quad k = 0, 1, \ldots, \left\lfloor \frac{N-1}{s} \right\rfloor,$$

The size is $(B, \lfloor(N-1)/s, C)$ for both.