TTIC 31230 Fundamentals of Deep Learning, winter 2019

CNN Problems

In these problems, as in the lecture notes, capital letter indeces are used to indicate subtensors (slices) so that, for example, $M[I,J]$ denotes a matrix while $M[i,j]$ denotes one element of the matrix, $M[i,J]$ denotes the ith row, and $M[I,j]$ denotes the jth collumn.

We also adopt the convention, similar to true Einstein notation, that repeated capital indeces in a product of tensors are implicitly summed. We can then write the inner product $e[w,I]^\top h[t,I]$ as $e[w,I]h[t,I]$. Using this implicit summation notation we can avoid ever using transpose.

Problem 1. Einstein Notation.
Suppose that at each time t from 1 to T we have a matrix $W_t \in \mathbb{R}^{IJ}$. Suppose we are given an initial vector x_0. We define the vectors $x_1, \ldots x_T$ by the equation
(1) $x_{t+1} = W_t x_t$
($W_t x_t$ is the matrix W_t times the vector x_t).
We can represent the vector x_t as a tensor $x[t,j]$ and represent the matrix W_t as a tensor $W[t,i,j]$.

part a: Rewrite equation (1) in terms of the tensors using summation notation.

part b: Rewrite equation (1) in terms of the tensors using Einstein notation where summation is represented by repeating a capital letter index.

Problem 2. Counting Floating Point Operations. Consider convolving a kernel $K[n_{\mathrm{out}}, \Delta x, \Delta y, n_{\mathrm{in}}]$ with thresholds $B[n_{\mathrm{out}}]$ on a layer $L[b,x,y,n_{\mathrm{in}}]$ where $B,\ X,\ Y, N_{\mathrm{out}}, N_{\mathrm{in}}, \Delta X,\ \Delta Y$ are the number of possible values for b, x, y, n_{out}, n_{in}, Δx and Δy respectively. How many floating point multiplies are required in computing the convolution on the batch (without any activation function)?

Solution:

$$BXY\ \Delta X\ \Delta Y\ N_{\mathrm{out}}\ N_{\mathrm{in}}$$

Problem 3. 3D Convolutions. Suppose that we want a video CNN producing layers of the form $L[b,x,y,t,n]$ which are the same as the layers of an image CNN but with an additional time index. Write the equation for computing $L_{\ell+1}[b,x,y,t,j]$ from the tensor $L_\ell[B,X,Y,T,I]$. Your filter should include an index Δt and handle a stride s applied to both space and time. Use the repeated index notation for summation.

Solution:

$$L_{\ell+1}[b,x,y,t,n_{\mathrm{out}}] = \sigma(K_{\ell+1}[n_{\mathrm{out}}\Delta X, \Delta Y, \Delta T, N_{\mathrm{in}}]L_\ell[b, sx{+}\Delta X, sy{+}\Delta Y, st{+}\Delta T, N_{\mathrm{in}}]{-}B[n_{\mathrm{out}}])$$

Problem 5. Incorporating Scale Invariance. Images have translation invariance — a person detector must look for people at various places in the image. Translation invariance is the motivation for convolution — all places in the image are treated the same.

Images also have some degree of scale invariance — a person detector must look for people of different sizes (near the camera or far from the camera). We would like to design a deep architecture that treats all scales (sizes) the same just as CNNs treat all places the same.

Consider a batch of input images $L_{0,d}[b,x,y,n]$ where $d = 2^k$ is the spacial dimension of x and y and n ranges over the three color values red, green, blue. To capture scale invariance will compute a set of layers $L_{\ell,d}$ with $0 \leq \ell \leq \ell_{\max}$ and d a power of 2 with $4 \leq d \leq d_{\max}$ where $d_{\max}$ is the spacial dimention of the input images. We set $d_{\min} = 4$ so as to allow 3×3 convolution kernels to be applied to the lowest spacial resolution. The output layer, say for image classificication, is $L_{\ell_{\max},d_{\min}}[b,x,y,n]$.

We first define $L_{0,d}[b,x,y,n]$ to be a layer in an "image pyramid" constructed by successively down-sampling the images by a factor 2.

$$L_{0,d/2}[b,x,y,n] \quad = \quad \frac{1}{4}\left(\begin{array}{l} L_{0,d}[b,2x,2y,n] + L_{0,d}[b,2x+1,2y,n] \\ +L_{0,d}[b,2x,2y+1,n] + L_{0,d}[b,2x+1,2y+1,n] \end{array}\right)$$

We next define $L_{\ell,d_{\max}}[b,x,y,n]$ by 3×3 convolutions that do not change the image dimension.

$$L_{\ell+1,d_{\max}}[b,x,y,n] = \sigma(K_{\ell+1}[n_{\mathrm{out}},\Delta X,\Delta Y,N_{\mathrm{in}}]L_{\ell,d_{\max}}[b,x+\Delta X,y+\Delta Y,N_{\mathrm{in}}] - B_{\ell+1}[n_{\mathrm{out}}])$$

For $d < d_{\max}$ give an equation for computing $L_{\ell+1,d}[b,x,y,n_{\mathrm{out}}]$ as the result of a linear threshold neuron taking inputs from both $L_{\ell,d}[b,x,y,n]$ **and** $L_{\ell,2d}[b,x,y,n]$ using the same kernel $K_{\ell+1}[n_{\mathrm{out}},\Delta x,\Delta Y,n_{\mathrm{in}}]$ for both inputs.

Solution:

$$L_{\ell+1,d}[b,x,y,n] = \sigma\left(\begin{array}{rl} & K_{\ell+1}[n_{\mathrm{out}},\Delta X,\Delta Y,N_{\mathrm{in}}]L_{\ell,d}[b,x+\Delta X,y+\Delta Y,N_{\mathrm{in}}] \\ + & K_{\ell+1}[n_{\mathrm{out}},\Delta X,\Delta Y,N_{\mathrm{in}}]L_{\ell,2d}[b,2x+\Delta X,2y+\Delta Y,N_{\mathrm{in}}] \\ - & B_{\ell+1}[n_{\mathrm{out}}] \end{array}\right)$$