MA 751
M. Kon

Problem Set 6
Due Thursday, 3/17/22

Lectures 11, 12

This problem set focuses on our discussion of convolutional neural networks, which are networks that 'scan' the feature vector input $\boldsymbol{x} = (x_1, \ldots, x_p)$ encoded into the first layer of a neural network and apply a simple repeating set of weights in the same way across the entire input layer (vector), computing simple local outputs across the second layer. This process then can be iterated across multiple layers. Deep convolutional networks developed by Geoff Hinton, et al.in 2012 began the shift back to neural nets as a fundamental and in fact transformative machine learning tool.

Reading: Class material

1. Learning differential calculus (G. Strang): The derivative process takes a polynomial $P(z) = a_0 + a_1 z + a_2 z^2 + a_3 z^3 + a_4 z^4$ to $P' = a_1 + 2a_2 z + 3a_3 z^2 + 4a_4 z^3$. Can you train a network to take derivatives by taking the *graph* of the polynomial P and teaching the network to output the constants a_1,$2a_2, 3a_3$, and $4a_4$?

Assume that you have trained a multilayer network with 4 output neurons, which take as the input vector $\boldsymbol{x} = (x_0, x_1, \ldots, x_{100})$ representing the values of $x_i = P(i/100)$, i.e. the samples of the values of the polynomial between 0 and 1. The four output neurons then compute the coefficients $a_1, 2a_2, 3a_3$, and $4a_4$, expressed as the activations of these four neurons.

What might such a network look like? That is, describe the architecture of a network that might accomplish this, including the number of layers and the network weights.

There is an implementation of such a network discussed in
https://blogs.mathworks.com/cleve/2018/08/06/teaching-calculus-to-a-deep-learner/

2. One dimensional filters. In class we have discussed how filters for one dimensional signals (e.g. pixels describing one row of a photograph) can serve as edge detectors, and gave an example of the filter $W = [-1/2,\ 0,\ 1/2]$. Recall that this filter represents the replacement of the original feature vector $\boldsymbol{x} = (x_1, \ldots, x_p)$ representing successive pixel intensities with a feature vector whose i^{th} entry is $-\frac{1}{2}x_{i-1} + \frac{1}{2}x_{i+1}$. We can always 'pad' $\boldsymbol{x}$ with zeroes at its ends so that these expressions also make sense there.

(a) Show that such a filter acts as a matrix of the form

$$A = -1/2R + 0I + 1/2L,$$

where R shifts the indices of the feature vector $\boldsymbol{x} = (x_1, \dots, x_p)$ to the right, L to the left and I leaves them alone. As mentioned, we 'pad' $\boldsymbol{x}$ with zeroes at the left end, so the right shift will produce a 0 in the first position, and we do the same on the right side {the purpose is to be sure that our operations are well-defined). What do R and L look like?
(b) Assuming our matrices are square and acting on $\mathbb{R}^6$, write out the matrices R and L (I is just the identity matrix).
(c) Now write out the matrix A. This is called a Toeplitz matrix - explain this definition.
(d) Assume again that $\boldsymbol{x}$ represents pixel intensities of a row in a photograph. Show that multiplying by the matrix A implements the filter W by replacing each pixel entry x_i as claimed above.
(e) How does this represent a derivative operation? How does multiplication by A act as an 'edge detector'?

3. Two dimensional filters. To consider a similar operation on two dimensional arrays of pixels, try the following. Instead of writing the array of pixel intensities as x_i with a single index i, try using a double indexing so that pixel x_{ij} is the j^{th} pixel in the i^{th} row. In this case it is easy to apply operations like left and right shifts (as well as up and down shifts) analogous to those we saw above.

Recall that it makes sense now to write down a filter such as

$$W = \begin{bmatrix} -1/2 & 0 & 1/2 \\ -1 & 0 & 1 \\ -1/2 & 0 & 1/2 \end{bmatrix}$$

which represents replacing each pixel x_{ij} by

$$-\frac{1}{2}\,x_{i-1,j+1} + \frac{1}{2}\,x_{i+1,j+1} - x_{i,j-1} + x_{i,j+1} - \frac{1}{2}\,x_{i-1,j-1} + \frac{1}{2}x_{i+1,j-1}.$$

(W is known as a *Sobel matrix*).

Note however that if we represent a two dimensional image as a standard feature vector (i.e. a single row vector) so as to perform standard matrix operations on it, we will need to use a *single* index i, representing our two dimensional image vector in the form $\boldsymbol{x} = (x_1, \dots, x_p)$. In order to find an equivalent matrix A that performs the same action as the filter W, proceed as follows.

(a) First, write down the filters W_i $(i = 1, \dots 4)$ representing respectively left shifts, right shifts, and upward and downward shifts.
(b) Now write down the matrix A_1 representing a left shift operation on our feature vector $\boldsymbol{x} = (x_1, \dots, x_p)$. Since this is a general question, simply indicate how A_1 should look based on the ordering of the features x_i representing pixels in the two dimensional image.
(c) Similarly explain in the same way how the matrices $A_2, A_3,$ and A_4 are constructed.

(d) Now write the matrix A_W that implements the above filter W as a matrix multiplication on the image vector $\boldsymbol{x}$. A matrix such as A_W is called a (generalized) Toeplitz matrix

4. Two dimensional edge detection. (a) If we view the matrix A_W above to be an edge detector, explain why it can detect edges in the original two dimensional image that $\boldsymbol{x} = (x_1, \ldots, x_p)$ represents. What kind of derivative does this operation represent in the original 2 dimensional pixel array? What kinds of edges can it detect?
(b) Try to express $\frac{1}{4}W$ above (a rank 1 matrix) as an outer product, i.e. a product of a column matrix times a row matrix. What form of this matrix product suggests a good representation of the derivative in the horizontal direction?
(c) Consider the matrix $W' = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$. How does it compare with W as a vertical edge detector (do not worry about possible normalization constants in front of these matrices).
(d) Suppose you wish to detect horizontal edges in your image. What would the corresponding filter V look like?
(e) What would be the corresponding matrix A_V?
(f) What happens if we wish to detect diagonal edges?