CSE616: Assignment (2)

1. Consider an input image of shape 500x500x3. The image is flattened and a fully connected layer with 100 hidden units is used. What is the shape of the weight matrix of this layer (without the bias)? What is the shape of the bias.

Shape of weight matrix = (500 * 500 * 3, 100) = (750000, 100)

Bias shape = (1, 100)

2. You run this image in a convolutional layer with 10 filters, of kernel size 5x5. How many parameters does this layer have?

Parameters = 5 * 5 * 3 * 10 = 750

3. The top gray image has run through different types of filters and the results are shown in the following images. What type of convolutional filter was used to get each of the resulting images. Explain briefly and include the values of these filters. The filters have a shape of (3,3).

Vertical edge = $\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ Horizontal edge = $\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$

4. In the Adam optimizer. Show what will happen the numbers of steps to compute the exponential moving averages gets large.

If the moving average gets large. the number of steps decreases

$$w_{t+1} = w_t - \frac{\alpha_t}{(v_t+\varepsilon)^{1/2}} * \left[\frac{\delta L}{\delta w_t}\right]$$

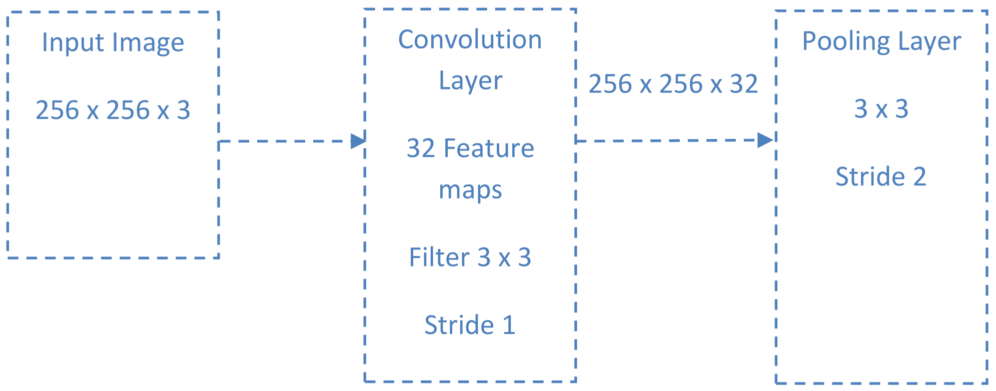
$$v_t = \beta v_{t-1} + (1-\beta) * \left[\frac{\delta L}{\delta w_t}\right]^2$$

5. Given a batch of size m, and assume that a batch normalization layer takes an input z = (z (1) , … , z (m)) as an input. Write down the output equation(s) of this layer. Give two reasons for using the batch normalization layer.

$$\hat{Z}(m) = \frac{z(m) - E[z(m)]}{\sqrt{Var[z(m)]}}$$

Using batch normalization makes the network more stable during training. This may require the use of much larger than normal learning rates, that in turn may further speed up the learning process.

6. Suppose you have a convolutional network with the following architecture: The input is an RGB image of size 256 x 256. The first layer is a convolution layer with 32 feature maps and filters of size 3x3. It uses a stride of 1, so it has the same width and height as the original image. _ The next layer is a pooling layer with a stride of 2 (so it reduces the size of each dimension by a factor of 2) and pooling groups of size 3 x 3. Determine the size of the receptive _eld for a single unit in the pooling layer. (i.e., determine the size of the region of the input image which influences the activation of that unit.) You may assume the receptive field lies entirely within the image.



Size of the receptive _eld for a single unit = 128 x 128

7. If an input data block in a convolutional network has dimension C×H×W = 96 ×128 ×128 , (96 channels, spatial dim 128x128) and we apply a convolutional filter to it of dimension D × C × HF × WF = 128×96 ×7×7 , (i.e. a block of D=128 filters) with stride 2 and pad 3,what is the dimension of the output data block?

$$W_{out} = \frac{W_{in} - K + 2P}{S} + 1$$
$$H_{out} = \frac{H_{in} - K + 2P}{S} + 1$$

W = (128 – 7 + 2 * 3) / 2 + 1 = 64

H = (128 – 7 + 2 * 3) / 2 + 1 = 64

the dimension of the output data block = 64 x 64 x 96 x 128

8. What is inverted dropout and what is its advantage?

Inverted dropout is a variant of the original dropout technique developed by Hinton et al. Just like traditional dropout, inverted dropout randomly keeps some weights and sets others to zero. This is known as the “keep probability” p.
The one difference is that, during the training of a neural network, inverted dropout scales the activations by the inverse of the keep probability q=1−pq=1−p. This prevents network’s activations from getting too large, and does not require any changes to the network during evaluation. In contrast, traditional dropout requires scaling to be implemented during the test phase.

9. Explain briefly why fully connected neural networks do not work well for image classification.

When the input image is of size (28x28x3) pixels, a fully connected neural network will have 2352 weights in the first hidden layer. In real life, the images have at least 200x200x3 pixels which results in 120,000 weights in the first hidden layer itself. Having so many parameters will result in overfitting. In such a scenario, we use convolutional neural networks.

10. Compute the convolution of the following two arrays: (4 1 -1 3) * (-2 1). Your answer should be an array of length 5. Show your detailed work.

Input: A[] = {4,1, -1, 3}, B[] = {-2, 1}

Output: {-8, 2, 3, -7, 3}

Explanation:

Size of array, C[] = N + M – 1 = 4 + 2 – 1 = 5.

	4	1	-1	3	
1	-2				
	1	-2			
		1	-2		
			1	-2	
				1	-2

C[0] = -2 * 4 = -8

C[1] = 1 * 4 - 2 * 1 = 2

C[2] = 1 * 1 - 2 * -1 = 3

C[3] = 1 * -1 - 2 * 3 = -7

C[4] = 1 * 3 = 3

11. Describe what setting change in epochs 25 and 60 could have produced this training curve. Be brief.

12. Why are convolutional layers more commonly used than fully-connected layers for image processing?

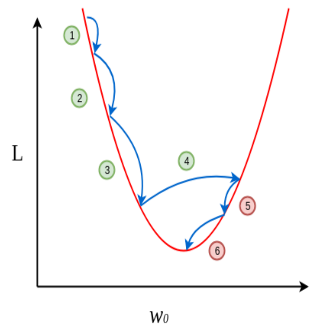
Convolutions are not densely connected, not all input nodes affect all output nodes. This gives convolutional layers more flexibility in learning. Moreover, the number of weights per layer is a lot smaller, which helps a lot with high-dimensional inputs such as image data.

13. Dropout layers implement different forward functions at train and test time. Explain what they do. Let p be the probability that node value is retained.

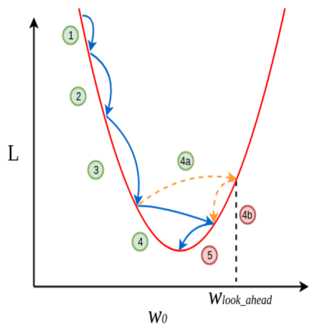
Dropout works by randomly and temporarily deleting neurons in the hidden layer during the training with probability p. We forward propagate input through this modified layer which has n ∗ p active neurons and also backpropagate the result through the same. During testing/prediction, we feed the input to unmodified layer, but scale the layer output with p.

14. Explain how standard momentum and Nesterov accelerated gradient differ. How does their performance (convergence as a function of time) differ on convex optimization problems? Use sketches as an aid.

The difference between Momentum method and Nesterov Accelerated Gradient is the gradient computation phase. In Momentum method, the gradient was calculated using current parameters. When the learning rate η is relatively large, Nesterov Accelerated Gradients allows larger decay rate α than Momentum method, while preventing oscillations. Both Momentum method and Nesterov Accelerated Gradient become equivalent when η is small.



(a) Momentum-Based Gradient Descent



(b) Nesterov Accelerated Gradient Descent

🟢 $\Longrightarrow \frac{\partial L}{\partial w_0} = \frac{Negative(-)}{Positive(+)}$ 🔴 $\Longrightarrow \frac{\partial L}{\partial w_0} = \frac{Negative(-)}{Negative(-)}$

15. How does the actual learning rate of ADAGRAD change with the number of steps?

In the above ADAGRAD optimizer equation, the learning rate has been modified in such a way that it will automatically decrease because the summation of the previous gradient square will always keep on increasing after every time step.

$$\text{Adagrad} \Rightarrow w_t = w_{t-1} - \eta_t' \frac{\partial L}{\partial w_{t-1}}$$

$$\text{where } \eta_t' = \frac{\eta}{\sqrt{\alpha_t + \varepsilon}}$$

ε is a small +ve number to avoid divisibilty by 0

$$\alpha_t = \sum_{i=1}^{t} \left(\frac{\partial L}{\partial w_{t-1}}\right)^2 \quad \textit{summation of gradient square}$$