Assignment #2

1) $500 \times 500 \times 3 = 750000$

- weight matrix $\longrightarrow$ 750000×100
- bias $\longrightarrow$ 1×100

---

2) 10 filters, kernel size 5×5

no of parameters $= 5 \times 5 \times 10 = 250 + 10$ bias $= 260$

---

3) - first image $\rightarrow$ vertical edge detection $\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$

- second image $\rightarrow$ horizontal edge detection $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

---

4) Adam optimizer makes use of the RMSProp optimization idea, which has the formula

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

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

So, we divide at each step by the total of previous gradients So, when the number of steps increase this means that the weight updates become smaller & the exponential moving average depends less on recent updates.

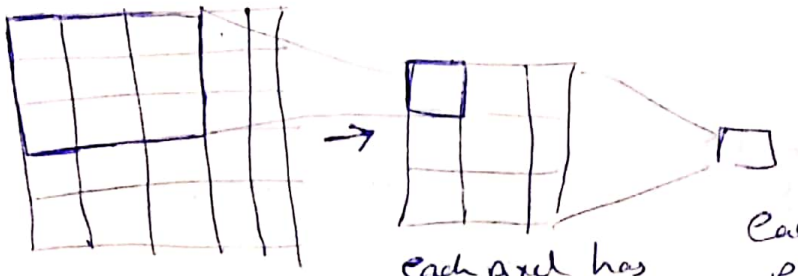
---

5) $\mu_z = \frac{1}{m} \sum_{i=1}^{m} z_i$, $\sigma_z^2 = \frac{1}{m} \sum_{i=1}^{m} (z_i - \mu_z)^2$

$$z_i = \frac{z_i - \mu_z}{\sqrt{\sigma_z^2}}$$

- It's used for faster & more robust training, as well as to reduce overfitting effect.

6)



Each pixel has receptive field of 3x3

Each pixel has receptive field 3x3

→ In Total, The receptive field of every unit of The pooling layer is 5x5

---

7)

- input → C x H x W = 96 x 128 x 128
- filters → D x C x HF x WF = 128 x 96 x 7 x 7

  S = 2 , P = 3
- output dimensions = $\frac{\text{i/p dim} - \text{kernelsize} + 2*P}{S} + 1$

  $= \frac{128 - 7 + 2*3}{2} + 1 = 64$
- dimension of output = 128 x 64 x 64

---

8) With dropout, we turn off some of the neurons with a probability (p), so to invert this effect, we have to scale the activations at test time with (1/p) to cancel the dropout effect

With inverted dropout, This scaling is done at training time, This leads to faster inference & Test Time

9) Because features in images are location invariant, This means That if the network learns a feature in some position of image it should be able to generalize the feature
Also, flattening an image to feed into fully connected network may mess up positional data of the image, and ruins the nature of image data.

---

10) • o/p size = $\frac{\text{i/p size} - \text{Kernel size} + 2P}{S} + 1$

$= \frac{4 - 2 + 2P}{1} + 1 = 5 \rightarrow \therefore$ Padding = 1

• Convolution = $(0, 4, 1, -1, 3, 0) * (-2, 1)$

$= (4, -7, -3, 5, -6)$

---

11) The learning rate is decreased (Learning Rate Decay)
This leads to the model taking smaller weight update steps & being able to reach the local minimum which couldn't be reach with the higher learning rate which can be shown from the flat Loss Curve

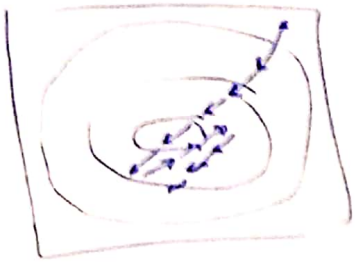
---

12) Convolutional filters are able to learn features available at any location of the image (location invariant) So they are more suitable for the nature of image data
Also with convolutional layer, less number of learnable parameters is needed to extract features from the image that what is need with FC layers.

3) • at train → p percentage of neurons are turned off
• at test → All neurons are on, so activations need to be scaled by factor of $(1/p)$

---

14) Standard momentum only takes in consideration recent updates when updating the parameters, this may lead to overshoots or noisy paths on the loss curve



With Nestrov accelerated momentum, a new term is taken into consideration which is a look ahead term, this helps for smoother optimization

---

15) Adagrad → $w_t = w_{t-1} - \eta_t \frac{\partial L}{\partial w_{t-1}}$

$$\eta_t = \frac{\eta}{\sqrt{\gamma_t + \epsilon}}$$

$$\gamma_t = \sum_{i=1}^{t} \left(\frac{\partial L}{\partial w_{i-1}}\right)^2$$