Puneet Mangla - CS17BTECH11029

Q.1 Initially there are 50% matches i.e $w = 0.5$
Homography has 8 degree of freedoms. so number of points to consider at a time $n = 4$

at k iterations, probability of bad model $= (1-w^n)^k$

To get 95% chance of exact homography means < 0.05 probability of bad model

$$(1-w^n)^k < 0.05$$
$$(1-(0.5)^4)^k < 0.05$$
$$k > \frac{\log(0.05)}{\log(1-(0.5)^4)}$$
$$k > 46.46$$

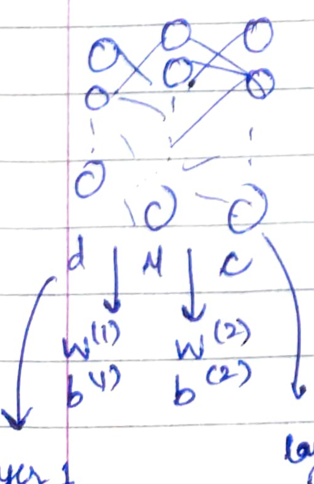
$k = 47$

Q.3

$$\Delta_{ij}^{(2)} := \Delta_{ij}^{(2)} + \delta_i^{(3)} * (a^{(2)})_j$$

$$\Delta^{(2)} := \Delta^{(2)} + \delta^{(3)} (a^{(2)})^T$$

Q.4



Total weights $= M \times d + c \times M$
Total biases $= M + C$.

$$\frac{\partial E}{\partial W^{(2)}} = \delta^{(3)} (a^{(2)})^T ; \quad \frac{\partial E}{\partial W^{(1)}} = \delta^{(2)} (a^{(1)})^T$$

we need to calculate only these.

so, Total independent derivatives $= M + C$

Q.2 $f(x) = \langle W^{(3)}, h^{(2)} \rangle = (W^{(3)})^T h^{(2)}$ {• means element wise product}

$$\frac{\partial f(x)}{\partial W^{(1)}_{ij}} = \frac{\partial (W^{(3)})^T h^{(2)}}{\partial W^{(1)}_{ij}} = (W^{(3)})^T \frac{\partial h^{(2)}}{\partial W^{(1)}_{ij}} \quad \left\{ h^{(2)} = \sigma(W^{(2)} h^{(1)}) \right.$$

$$\frac{\partial f(x)}{\partial W^{(1)}_{ij}} = (W^{(3)})^T \left[h^{(2)} \cdot (1-h^{(2)}) \cdot \frac{\partial W^{(2)} h^{(1)}}{\partial W^{(1)}_{ij}} \right] \quad \left\{ \frac{d\sigma(x)}{dx} = \sigma(x)(1-\sigma(x)) \right.$$

$$\frac{\partial f(x)}{\partial W^{(1)}_{ij}} = (W^{(3)})^T \left[h^{(2)} (1-h^{(2)}) \left[W^{(2)} \frac{\partial h^{(1)}}{\partial W^{(1)}_{ij}} \right]\right] \quad \left\{ h^{(1)} = \sigma(W^{(1)} x) \right.$$

$$\frac{\partial f(x)}{\partial W^{(1)}_{ij}} = (W^{(3)})^T \left[h^{(2)} \cdot (1-h^{(2)}) \cdot \left[W^{(2)} \left[h^{(1)} \cdot (1-h^{(1)}) \cdot \frac{\partial W^{(1)} x}{\partial W^{(1)}_{ij}} \right]\right]\right]$$

$$\frac{\partial f(x)}{\partial W^{(1)}_{11}} = (W^{(3)})^T \left[h^{(2)} \cdot (1-h^{(2)}) \cdot \left[W^{(2)} \begin{bmatrix} h^{(1)}_1 (1-h^{(1)}_1) x_1 \\ 0 \end{bmatrix} \right]\right]$$

$$\frac{\partial f(x)}{\partial W^{(1)}_{12}} = (W^{(3)})^T \left[h^{(2)} \cdot (1-h^{(2)}) \cdot \left[W^{(2)} \begin{bmatrix} h^{(1)}_1 (1-h^{(1)}_1) x_2 \\ 0 \end{bmatrix} \right]\right]$$

$$\frac{\partial f(x)}{\partial W^{(1)}_{21}} = (W^{(3)})^T \left[h^{(2)} \cdot (1-h^{(2)}) \cdot \left[W^{(2)} \begin{bmatrix} 0 \\ h^{(1)}_2 (1-h^{(1)}_2) x_1 \end{bmatrix} \right]\right]$$

$$\frac{\partial f(x)}{\partial W^{(1)}_{22}} = (W^{(3)})^T \left[h^{(2)} \cdot (1-h^{(2)}) \cdot \left[W^{(2)} \begin{bmatrix} 0 \\ h^{(1)}_2 (1-h^{(1)}_2) x_2 \end{bmatrix} \right]\right]$$

Q.5 $\quad y_n = f(x_n, w) + \varepsilon_n$

$\Rightarrow \quad y_n \sim N(f(x_n, w), \Sigma)$

$$P(y_n = y \mid x_n) \propto e^{-(y-f(x_n,w))^T \Sigma^{-1} (y - f(x_n,w))}$$

Now $\quad P(Y \mid X) = \prod_{i=1}^{N} P(y_i \mid x_i)$

$$\propto e^{-\sum_{i=1}^{N} (y_i - f(x_i, w))^T \Sigma^{-1} (y_i - f(x_i,w))}$$

Loss is negative log likehihood.

$$\mathcal{L}(w) = -\log P(Y \mid X)$$

$$= \sum_{i=1}^{N} (y_i - f(x_i, w))^T \Sigma^{-1} (y_i - f(x_i, w))$$

$$w^* = \underset{w}{\operatorname{argmin}} \ \mathcal{L}(w).$$

Σ is p.s.d ~~so~~ can be written as $S \Lambda S^T$ [diagonalization]

$$\Rightarrow \mathcal{L}(w) = \sum_{i=1}^{N} (y_i - f(x_i,w))^T S^{-T} \Lambda^{-1} S^{-1} (y_i - f(x_i,w))$$

$$= \sum_{i=1}^{N} (S^{-1} y_i - S^{-1} f(x_i,w))^T \Lambda^{-1} (S^{-1} y_i - S^{-1} f(x_i,w))$$

$$= \sum_{i=1}^{N} (y_i' - f'(x_i,w))^T \Lambda^{-1} (y_i' - f'(x_i,w))$$

here $y_i' = S^{-1} y_i$ and $f'(x_i, w) = S^{-1} f(x_i, w)$

so the problem becomes simple regression. i.e

$$y_n' = f'(x_n, w) + \varepsilon_n' \qquad \varepsilon_n' \sim N(0, \Lambda)$$

$$\Lambda = \operatorname{diag}[\sigma_1^2, \sigma_2^2 \cdots \sigma_d^2]$$

Q.6 a) Let $\mathcal{L}(\mathcal{D}, w_1, w_2)$ be the loss function of neural network on dataset $\mathcal{D}$ with two weights w_1 and w_2.

Scale-symmetry exists ie

$$\mathcal{L}(\mathcal{D}, w_1, w_2) = \mathcal{L}(\mathcal{D}, \gamma w_1, \frac{1}{\gamma} w_2) \quad - \text{Eq-(1)}$$

~~$\frac{\partial \mathcal{L}(\mathcal{D}, W_1, W_2)}{\partial}$~~ Now, taking / calculating gradients for updates.

$$\frac{\partial \mathcal{L}(\mathcal{D}, \gamma w_1, \frac{1}{\gamma} W_2)}{\partial W_1} = \frac{\partial \mathcal{L}(\mathcal{D}, \gamma w_1, \frac{1}{\gamma} W_2)}{\partial (\gamma W_1)} \cdot \frac{\partial (\gamma w_1)}{\partial W_1}$$

$$= \frac{\partial \mathcal{L}(\mathcal{D}, W_1, W_2)}{\partial (W_1)} \cdot \gamma$$

$\{ \because$ using Eq-(1) $\}$

Similarly,

$$\frac{\partial \mathcal{L}(\mathcal{D}, \gamma W_1, \frac{1}{\gamma} W_2)}{\partial W_2} = \frac{\partial \mathcal{L}(\mathcal{D}, W_1, W_2)}{\partial W_2} \cdot \frac{1}{\gamma}$$

Hence gradients w.r.t to W_1, and W_2 are also scaled with γ and $\frac{1}{\gamma}$ respectively.

~~If~~ This will create uneven updates as if γ is high W_1 gradient will explode and W_2 gradient will vanish. ~~and~~ Similarly if γ is low W_1 grad. can vanish and W_2 will explode.

This will increase instability of training as updates will be uneven.

b) Permutation Symmetry:

In any Multi layered network ~~&~~ with d layers and n neurons. There will be $(n!)^d$ permutations of hidden units.

Given any permutation, we can find weight configuration that gives same output. ~~Sin~~ And it is more-likely these configurations are local-minima. Thus ~~bot~~ we could encounter many local minimas during training. We need to initialize network carefully for better performance.