Assignment 7

Tuesday, 3. January 2023 22:27

Prathrish Mithare - 7028692
Subrat Kishore Dutta - 7028082

Exercise 7.1

a] From the paper we get to know that a feedforward network with given hidden layer, with given number of neurons can approximate continuous functions under appropriate activation functions. And also from the paper we get to know the reason why neural network perform well and can approximate functions.

b] The main idea of the paper from part a is that since it uses appropriate activation function with given hidden layer it can approximate any continuous function. Approximation by superposition of a sigmoidal function uses sigmoid as an activation function, thus we can say activation function plays an important role.

c] Deeper networks can learn more complex relationships in the data and also they can generalize better which thus reduces overfitting. Deeper networks can also learn the features [ex: edges] better.

Exercise 7.2 L_1 and L_2 Regularization

(a) The bias term is responsible for only controlling one variable on the other hand weights control two variables i.e it specifies how two variable interacts. The bias term do not contribute much to the curvature of the surface that the model learns, thus regularising it does not reduce the overfitting (high variance) or in other words does not induce much bias. On the contrary it can introduce significant underfitting as the position of the surface along the corresponding dimension can shift significantly.

(b) As a result of using L_1 regularisation the solution we obtain is sparse meaning, the coefficients corresponding to a subset of the parameters can turn out to be zero. Using such an approach can be advantageous when we want a simple model which only takes into consideration the most important features in a dataset.

L_2 regularisation although penalises high weight values and attempts to bring them close to zero provides more flexibility as it still allows them. This characteristic is highly desirable when every feature can be responsible for the model's performance doe eliminating one of them can be catastrophic. It is particularly important when the features are correlated and their interaction can result in specific pattern

(c) It is possible to use both L_1 and L_2 regularisation within a loss function as a regularisation term. They can be used in a linear combination with appropriate weight assigned to them such that a balance is maintained between the two.

Using them together in some weighted combination can be beneficial as they will create a balance between the sparsity offered by L_1 regularisation and the flexibility offered by L_2 regularisation.

(d) In general it is true as regularisation is used to prevent overfitting and make a balance between minimising the training error and minimizing the generalization error. When we regularise by adding penalty it normally has a tendency to increase the training error but it results in the model not overfitting and generalizing better resulting in better performance on the test data.

(e) The penalty term added to the objective function is such that it penalizes the model if the weights are too large. The optimization algorithm will try to minimise the objective function by reducing the weights of the model, thus if the weight becomes too large the optimization algorithm may turn that to zero.

e.g.

the weight is updated as

$$w_{new} : w_{old} - \alpha \frac{\partial L_1}{\partial w}$$

taking $\alpha = 0.5$ and $w_{old} = 1$

upon updating

$$w_{t+1} = 1 - 0.5(1) = 0.5.$$

$$w_{t+2} = 0.5 - 0.5(1) = 0,$$

as we can see here the weight can be set to zero.

in case of L_2 with only one feature and in case of a L_2 regularised least square case the loss function can be approximated around the optimal value w^*

$$\hat{J}(w) = J(w^*) + \frac{1}{2}(w-w^*)H(w-w^*)$$

taking its derivative gives we get 0 at minima

$$\frac{\partial \hat{J}(w)}{\partial w} = 0 + H(w-w^*) = 0$$

$$\Rightarrow \frac{\partial \hat{J}(w)}{\partial w} = H(w-w^*) = 0 \longrightarrow (1)$$

now the derivative of L_2 regularisation gives αw adding this to eq^n (1) and at minima.

$$\Rightarrow H(w-w^*) + \alpha w = 0$$

$$\Rightarrow Hw - Hw^* + \alpha w = 0$$

$$\Rightarrow (H+\alpha)w = Hw^*$$

$$\Rightarrow w = \left(\frac{H}{H+\alpha}\right)w^*$$

now in a least square problem $H > 0$ hence unless w^* is 0 w cannot be zero.

Exercise 7.3:

a] $w_{i+1} = w_i - \eta \nabla_w J(w) \quad — (1)$

For regularised loss the equation (1) becomes

$$w_{i+1} = w_i - \eta(\nabla_{w_i} J(w_i) + \lambda w_i)$$

$$= w_i - \eta\lambda w_i - \eta \nabla_{w_i} J(w_i)$$

$$= w_i(1-\eta\lambda) - \eta \nabla_{w_i} J(w_i)$$

where η : learning parameter

λ = regularization parameter.

$(1-\eta\lambda)$ = weight decay.

b] $\tilde{w} = (H+\alpha I)^{-1} H w^*$

$$\hat{J}(w) = J(w^*) + \frac{1}{2}(w-w^*)^T H (w-w^*)$$

$$\nabla_w \hat{J}(w) = H(w-w^*)$$

$$w^{(T)} = w^{(T-1)} - \varepsilon \nabla_w \hat{J}(w^{(T-1)})$$

$$= w^{(T-1)} - \varepsilon H(w^{(T-1)} - w^*)$$

$$w^{(T)} - w^* = w^{(T-1)} - \varepsilon H w^{(T-1)} + \varepsilon H w^* - w^*$$

By taking common elements

$$w^{(T)} - w^* = (I - \varepsilon H)(w^{(T-1)} - w^*)$$

We know that $H = Q\Lambda Q^T$

$$w^{(T)} - w^* = (I - \varepsilon Q\Lambda Q^T)(w^{(T-1)} - w^*)$$

$$Q^T(w^{(T)} - w^*) = (IQ^T - \varepsilon I\Lambda Q^T)(w^{(T-1)} - w^*)$$

$$Q^T(w^{(T)} - w^*) = (Q^T - \varepsilon\Lambda Q^T)(w^{(T-1)} - w^*)$$

$$Q^T(w^{(T)} - w^*) = (I - \varepsilon\Lambda)Q^T(w^{(T-1)} - w^*)$$

$$Q^T w^{(T)} - Q^T w^* = (I - \varepsilon\Lambda)Q^T(w^{(T-1)} - w^*)$$

ε is chosen small & initial weight is zero

$$Q^T w^{(T)} - Q^T w^* = (I - \varepsilon\Lambda)^T Q^T(-w^*)$$

$$Q^T w^{(T)} = Q^T w^* + (I - \varepsilon\Lambda)^T Q^T(-w^*)$$

$$Q^T w^{(T)} = Q^T w^*[I - (I - \varepsilon\Lambda)^T] \longrightarrow (1)$$

After L2 norm we get

$$\tilde{w} = (H + \alpha I)^{-1} H w^*$$

$$= (Q\Lambda Q^T + \alpha I)^{-1} Q\Lambda Q^T w^*$$

$$= [Q(\Lambda + \alpha I)Q^T]^{-1} Q\Lambda Q^T w^*$$

$$= Q(\Lambda + \alpha I)^{-1}\Lambda Q^T Q\Lambda Q^T w^*$$

$$= Q(\Lambda + \alpha I)^{-1}\Lambda Q^T w^*$$

$$Q^T\tilde{w} = Q^T Q(\Lambda + \alpha I)^{-1}\Lambda Q^T w^*$$

$$= (\Lambda + \alpha I)^{-1}\Lambda Q^T w^*$$

this can be written as

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1} V A^{-1}$$

$U = 1$, $C = \alpha I$, $V = 1$

$$(A + \alpha I)^{-1} = \Lambda^{-1} - \Lambda^{-1}(I/\alpha + \Lambda^{-1})^{-1}\Lambda^{-1}$$

$$(\Lambda + \alpha I)^{-1}\Lambda = \Lambda^{-1}\Lambda - \Lambda^{-1}(I/\alpha + \Lambda^{-1})^{-1}\Lambda^{-1}\Lambda$$

$$= I - \Lambda^{-1}(I/\alpha + \Lambda^{-1})^{-1}$$

$$(AB)^{-1} = B^{-1}A^{-1}$$

$$= \Lambda^{-1}(I/\alpha + \Lambda^{-1})^{-1}$$

$B = \Lambda$

$A = (I/\alpha + \Lambda^{-1})$

$$(B^{-1}A^{-1}) = (AB)^{-1}$$

$$= ((I/\alpha + \Lambda^{-1})\Lambda)^{-1}$$

$$= \left(\frac{\Lambda}{\alpha} + 1\right)^{-1}$$

$$= \left(\frac{1}{\alpha}(\Lambda + \alpha I)\right)^{-1}$$

Thus

$$(\Lambda + \alpha I)^{-1}\Lambda = I - \alpha(\Lambda + \alpha I)^{-1}$$

$$Q^T\tilde{w} = [I - (\Lambda + \alpha I)^{-1}\alpha]Q^T w^* \quad —— (2)$$

Comparing (1) & (2)

$$(I - \varepsilon\Lambda)^T = (\Lambda + \alpha I)^{-1}\alpha$$

By taking log on both sides & using series approximation of $\log(1+x)$

$$T \approx \frac{1}{\varepsilon\alpha}$$

Increasing T reduces regularization constant & vice versa.

c] $P(y|x_i) = \begin{cases} 1 & \text{if } y = y_i \\ 0 & \text{else} \end{cases}$

$$L = \sum_{i=1}^{n} H_i(P, q_\theta)$$

$$= -\sum_{i=1}^{n}\sum_{y=1}^{k} P(y|x_i)\log q_\theta(y|x_i)$$

$$= -\sum_{i=1}^{n} \log(q_\theta(y|x_i))$$

This is minimizing the loss function equal to maximum likelihood

$$q_\theta(y_i|x_i) = \frac{\exp(Z_{y_i})}{\sum_{j=1}^{k}\exp(Z_j)}$$

This will lead to overfitting, thus noise is added to one hot encoding

$$\therefore P'(y|x_i) = (1-\varepsilon)P(y|x_i) + \varepsilon U(y|x_i)$$

$$= \begin{cases} (1-\varepsilon) + \varepsilon U(y|x_i) & y = y_i \\ \varepsilon U(y|x_i) & \text{else} \end{cases}$$

Thus [0,1,0,0] might change to [0.1,0.7,0.1,0.1]

Exercise 7.4 : Dropout

(a) Dropout is a regularization technique that is used in a network where certain fraction of the nodes in a layer are turned off randomly during the training period. Dropout prevents the model from being dependent on specific nodes and basing the decision entirely on those nodes, instead it makes the model to learn corresponding to each node thereby extracting more generalised features from the data.

When the model is shallow and has less capacity using dropout may result in underfitting as the model may not be able extract any proper information.

(b) Bagging involves training multiple models on the same data and combines the output from all them, dropout however is done within the same model within a single training loop.

The way Bagging and dropout is similar is as dropout tries to mimic bagging. As in dropout different node is turned off in each iteration of training, in each iteration a subset of the model is trained. This mimics training different models and the resultant model gives a value which is like an ensembled product of all the submodels trained.

(c) We can apply both forms of regularisation technique to a neural network. This results in better generalisation and prevents overfitting

The study by Srivastava et. al. (2014) showed how combination of dropout with other regularisation technique like L_2 and Max-norm gives better results than when used individually.

(d) In dropout, a certain percentage (p) of neurons are set to zero and the rest of the units are scaled down by a factor $\frac{1}{1-p}$.

In inverted dropout the input units are scaled up by a factor of $1/(1-p)$. The dropping of the units is done during the inference time using a dropout mask randomly.