UNIT 3

PROBABILISTIC MODELING AND NEURAL NETWORKS

SYLLABUS

Probabilistic Modeling: Classification by Density Estimation, Statistical Estimation, Naive Bayes Models, Prediction.

Neural Networks: Bio-inspired Multi-layer Networks, The Back-propagation Algorithm, Initialization and Convergence of Neural Networks, Beyond Two Layers, Breadth vs Depth, Basis Functions.

LEARNING OBJECTIVES

- Brief Introduction to Probabilistic Modeling
- Density Estimation and Statistical Estimation
- Naive Bayes Model and its Predictions
- Introduction to Neural Networks
- Back Propagation and its Algorithm
- Initialization and Convergence of Neural Networks
- Versions of Algorithms Beyond Two Layers
- Introduction to Deep Networks.

INTRODUCTION

Probabilistic modeling can be defined as a modern approach that predicts the probability of future results. Naive Bayes Model can be defined as the various algorithms whose main objective is to create and provide a label to the data based on features or inputs provided by users.

A Neural Network (NN) is a large parallel distributed processor made-up of simple processing 'nodes' or units and connections from one node to another. These units are equivalent to 'neuron' inside a human brain and hence also called as "Neurons" in the network. These are the essential units for processing information throughout the system.

Back propagation network is the most well-known and the widely used technique of teaching artificial neural networks about how to do a particular task. It uses supervised learning with more powerful learning rule to implement gradient descent in weight space for a multilayer feed forward network. This learning rule is known as 'back propagation'.

SHORT QUESTIONS AND ANSWERS

Q1. Define density estimation.

Model Paper

Answer :

Density estimation can be defined as a method of determining the probability density function of a r variable from the given population.

The different density estimation approaches include the parametric methods (Like Gaussian or N distribution model) and non-parametric methods (like Kernel Density Estimation).

Q2. What is statistical estimation?

Answer :

Statistical estimation can be defined as an approach that determines the parameters of a statistical m using the sample data. The main objective of this approach is to obtain the best values that fit the data.

Consider an example of flipping a coin that is possibly biased, and the observed data is, *HHTH*.

Here, *H* represents the outcome as Heads

T represents the outcome as Tails

The outcome in each flip is independent, eventually it is considered as i.i.d.

Q3. Define Naive Bayes Model with an example.

Answer :

Model Paper-I

Naive Bayes Model

Naive bayes models can be defined as the various algorithms whose main objective is to create and pr a label to the data based on the features or inputs provided by the users.

Example

Labeling a movie as hit or flop based on the words used in the review. Hence, the probability for a point data can be given as,

$$P_\theta((y, x)) = P_\theta(y, x_1, x_2, \ldots, x_D)$$

Q4. Explain prediction with example.

Answer :

Predication

(Model Paper-I, Q7 | June/Aug.-22, Q7

Prediction can be defined as an outcome of an algorithm/model that has been trained based on pas historical datasets. It is then applied to test data inorder to forecast the probability of specific outcome.

Example

In business, prediction can be used to predict the future trends and identify the patterns based on pas These trends and patterns enables the business/enterprises to take better decisions and handle the tasks s financial and marketing operations, future production.

Q5. What are Bio-inspired Multi-layer Networks?

Answer :

(Model Paper-II, Q8 | July/Aug.-22, Q3 (M

A multilayer network can be defined as a network that contain input layer, output layer and one or hidden layer. The input layer supply data to the network. The output neurons and the hidden neurons perfo processing of information in the network. The network with hidden neurons acquire global perspective w function is to extract higher order statistics and provide some connectivity between the input units and output The input layer is directed towards the hidden layer which in turn is directed towards its next layer and finall last layer connects to the output layer which gives the final result. This shows the concept of two or more la hence the name multilayer networks.

Q6. Write an algorithm for two layer network prediction using hyperbolic tangent function.

Answer :

Model Paper-I, Q8

The two-layer network prediction algorithm by using hyperbolic tangent function can be given as,

Algorithm (TwoLayerNetworkPredict(W, v, $\hat{x}$))

Inputs : W, v, $\hat{x}$

Output : $v.h$

Algorithm : for i = 1 to number-of-hidden-units do

$h_i \leftarrow \tanh(w_i . \hat{x})$

end for

return $v.h$

Q7. State the two-layer network theorem.

Answer :

Statement

The two-layer networks are universal function approximators.

(or)

The two-layer networks can approximate any function.

Explanation

Let F be a continuous function on a bounded subset of D-dimensional space. Then there exists a two-layer neural network $\hat{F}$ with a finite number of hidden units that approximate F arbitrarily well. Namely, for all x in the domain of F, $| F(x) - \hat{F}(x) | < \epsilon$.

Q8. State some of the applications of back propagation algorithm.

Answer :

(Model Paper-III, Q8 | June/Aug.-22, Q9 [Old])

Some the applications of back propagation algorithm are,

1. Face recognition
2. Signature verification
3. Speech synthesis
4. System identification
5. Character recognition.

Q9. Write in brief about forward propagation algorithm.

Answer :

Model Paper-I, Q9

The forward propagation algorithm is used to calculate the activation of the sink y with given inputs. In this algorithm, initially all the activations for the parent nodes of u are calculated and then the activations for the node 'u' will be calculated. This algorithm goes deeper till the child nodes.

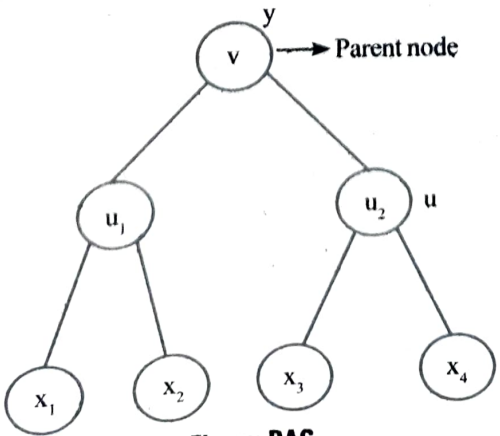

Figure: DAG

Q10. Explain the layers of RBF function.

Answer : Model Paper-II, Q

The RBF network consists of three different layers. They are as follows,

1. Input Layer

It consists of various nodes that creates connection between the network and its environment.

2. Hidden Layer

It is a single layer a network and performs non-linear transformations from input space to hidden space.

3. Output Layer

It consists of various linear nodes that are responsible for sending the results of the network to activation pattern.

Q11. Explain about issues in parity function.

Answer : (Model Paper-III, Q9 | June/Aug.-22, Q8 [OU])

Parity function is a generalization of XOR problem. It can be defined as,

$$\text{parity}(x) = \sum_d x_d \bmod 2 \begin{cases} \rightarrow 1 \text{ when number of 1s in x is odd} \\ \rightarrow 0 \text{ when number of 1s in x is even} \end{cases}$$

Parity function requires inspection of every feature for making predictions. It involves strongest inter class dependencies in classification problems making it difficult to obtain equivalences. The problems become more complex when number of attributes increases. However, there are certain algorithms that can overcome these issues.

PART - B

ESSAY QUESTIONS AND ANSWERS

3.1 PROBABILISTIC MODELING

3.1.1 Classification by Density Estimation

Q12. Define probabilistic Modeling and explain about classification by density estimation.

Answer :

Probabilistic Modeling

Probabilistic modeling can be defined as a modern approach that predicts the probability of future results by using the effect of random actions.

Classification by Density Estimation

Density estimation can be defined as a method of determining the probability density function of a random variable from the given population.

The Bayes optimal classifier with the distribution D is given as,

$$f^{(BO)}(\hat{x}) = \arg\max_{\hat{y} \in Y} D(\hat{x}, \hat{y})$$

But, in this approach the distribution is unknown and the optimality of this approach allows the users to determine $\hat{D}$ which is similar to D. Eventually, this $\hat{D}$ can be used as distribution in the classification.

The different density estimation approaches include the parametric methods (Like Gaussion or Normal distribution model) and non-parmetric methods (like Kernel Density Estimation).

The most efficient and suggested way to construct the probability distribution is to use the parametric methods which includes mean and variance as its parameters. The main objective of this learning is to get the best parameters from the observed training data. Note as assumption that, whenever a sample is drawn it should be drawn independently from D and should not depend on the previous samples. For instance, the samples (x_1, y), (x_2, y_2), ... $(x_n, y_n) \in D$.

Thus, with this assumption one can conclude that the samples drawn from the training data is i.i.d which means independently and identically distributed. This assumption is considered as a key assumption in machine learning.

3.1.2 Statistical Estimation

Q13. Explain about statistical estimation.

Answer :

Statistical estimation can be defined as an approach that determines the parameters of a statistical model by using the sample data. The main objective of this approach is to obtain the best values that fit the data.

Consider an example of flipping a coin that is possibly biased, and the observed data is, *HHTH*.

Here, *H* represents the outcome as Heads,

T represents the outcome as Tails and

The outcome in each flip is independent, eventually it is considered as i.i.d.

Let the probability of occurring heads is 'β', thus probability of occurring tails is "1 – β". Now, perform the maximum likelihood estimation, by using the scalar parameter β. Thus, the probability can be computed as,

$$P_\beta(D) = P_\beta(HHTH)$$

$$= P_\beta(H).P_\beta(H).P_\beta(T).P_\beta(H) \qquad [\because \text{ Each flip is independent}]$$

$$= \beta.\beta.(1-\beta).\beta.$$

$$= \beta^3(1-\beta)$$

$$P_\beta(D) = \beta^3 - \beta^4$$

Inorder to get the value of 'β' that maximizes the probability of data, perform the derivative of $P_\beta(D)$ with respect to β and equal it to '0'.

That is,

$$\frac{\partial}{\partial\beta}(\beta^3 - \beta^4) = 0$$

$$\Rightarrow \quad 3\beta^2 - 4\beta^3 = 0$$

$$\Rightarrow \quad \beta^2(3 - 4\beta) = 0$$

$$\Rightarrow \quad 3 - 4\beta = 0$$

$$\Rightarrow \quad 4\beta = 3$$

$$\Rightarrow \quad \beta = \frac{3}{4} = 0.75$$

In general, this problem can be solved for H heads and T tails by using the below formula,

$$\beta^H(1-\beta)^T$$

Now, use and derivate the above formula w.r.t β and equal it to zero.

Alternate Approach

The log likelihood or log probability is considered as an alternate and friendly approach to obtain the value of 'β' that maximizes the probability of data. The sequence can be given as,

$$\Rightarrow \quad H \log\beta + T \log(1-\beta)$$

Now, differentiate the above equation with 'β' and assign it to '0'.

$$\frac{\partial}{\partial\beta}[H \log\beta + T.\log(1-\beta)] = 0$$

$$\Rightarrow \quad H.\frac{\partial}{\partial\beta}(\log\beta) + T.\frac{\partial}{\partial\beta}(\log(1-\beta)) = 0$$

$$\Rightarrow \quad \frac{H}{\beta} - \frac{T}{(1-\beta)} = 0$$

$$\Rightarrow \quad \frac{H}{\beta} = \frac{T}{1-\beta}$$

$$\Rightarrow \quad H - H\beta = T.\beta$$

$$\Rightarrow \quad T\beta + H\beta = H$$

$$\Rightarrow \quad \beta(H+T) = H$$

$$\therefore \quad \beta = \frac{H}{H+T}$$

classification.

Answer :

Probabilistic Modeling

(Model Paper-I, Q15(a) | June/Aug.-22, Q15(a) [OU])

For answer refer Unit-III, Page No. 57, Q.No. 12, Topic: Probabilistic Modeling.

Estimation Methods for Classification

The different estimation methods used for classification are,

1. Density estimation
2. Statistical estimation.

1. **Density Estimation**

For answer refer Unit-III, Page No. 57, Q.No. 12, Topic: Classification by Density Estimation.

2. **Statistical Estimation**

For answer refer Unit-III, Page No. 57, Q.No. 13.

3.1.3 Naive Bayes Models

Q15. Write in detail about Naive Bayes models.

Answer :

Naive bayes models can be defined as the various algorithms whose main objective is to create and provide a label to the data based on the features or inputs provided by the users.

Example

Labeling a movie as hit or flop based on the words used in the review. Hence, the probability for a single point data can be given as,

$$P_\theta((y, x)) = P_\theta(y, x_1, x_2, \ldots, x_D) \quad \ldots (1)$$

As the above equation consists of distribution over many variables, it is complex to work with the probability distribution. Hence, this can be simplified using the chain rule of probabilities as given below,

$$P_\theta(x_1, x_2, \ldots, x_D, y) = P_\theta(y)\, P_\theta(x_1 | y)\, P_\theta(x_2 | y, x_1)\, P_\theta(x_3 | y, x_1, x_2) \ldots P_\theta(x_D | y, x_1, x_2, x_3, \ldots, x_{D-1})$$

$$= P_\theta(y) \prod_d P_\theta(x_d | y, x_1, \ldots, x_{d-1}) \quad \ldots (2)$$

Inorder to ignore the complexity in computing distribution over large number of features, make an assumption called as "naive bayes assumption". It states that "The features are independent, conditioned on the label". For instance, if the review of a movie is positive then the probability of a word "Excellent" is independent of other words such as "blockbuster". Mathematically, this assumption can be written as,

$$P(x_d | y, x_{d'}) = P(x_d | y), \forall d \neq d' \quad \ldots (3)$$

Using equation (3), the equation (2) can be simplified as,

$$P_\theta((y, x)) = P_\theta(y) \prod_d P_\theta(x_d | y) \quad \ldots (4)$$

From equation (4), the value of P can be parameterized. For instance, consider the labels as well as the features are binary. Hence, in this case, model the label as a biased coin having the probability of heads (say, positive review) will be given by θ_0. Further, assume the each label has a biased coin, for one feature. Ultimately, there will be $1 + 2D$ coins for D features, where one represents the labels (θ_0) and the other represents the combination of label/features (θ_{+1} and θ_{-1}). Therefore, the log probability can be written as,

$$P_\theta((y, x)) = P_\theta(y) \prod_d P_\theta(x_d \mid y)$$

$$= (\theta_0^{[y=+1]} (1-\theta_0)^{[y=-1]}) \prod_d \theta_{(y),d}^{[x_d=1]} (1-\theta_{(y),d})^{[x_d=0]}$$

Solving the above equation for θ_0 is same as the computation for biased coin. Hence, the relative frequency for the positive labels in the given data can be given as,

$$\hat{\theta}_0 = \frac{1}{N} \sum_n [y_n = +1]$$

$$\hat{\theta}_{(+1),d} = \frac{\sum_n [y_n = +1 \wedge x_{n,d} = 1]}{\sum_n [y_n = +1]}$$

$$\hat{\theta}_{(-1),d} = \frac{\sum_n [y_n = -1 \wedge x_{n,d} = 1]}{\sum_n [y_n = -1]}$$

Now that, use another model i.e., Bernoulli distribution when the features are not binary. And, use distribution or Gaussian distribution when the data is continuous.

Q16. Discuss about Classification by Density Estimation and Naive Bayes Models.

Answer :

(Model Paper-II, Q15(a) | July/Aug.-22, Q1

Classification by Density Estimation

For answer refer Unit-III, Page No. 57, Q.No. 12, Topic: Classification by Density Estimation.

Naive Bayes Model

For answer refer Unit-III, Page No. 59, Q.No. 15.

3.1.4 Prediction

Q17. Write short note on simplifying predictions made by Naive Bayes model.

Answer :

The predictions made by naive bayes model with bernoullis features can be represented as,

$$P_\theta((y, x)) = P_\theta(y) \prod_d P_\theta(x_d \mid y)$$

This can be mode simple by considering the decision boundaries. The decision boundaries may vary from one model to another model. Incase of probabilistic models, the decision boundary is precisely 50% likelihood of its inputs i.e., $y + 1$ is precised by 50%. This can also be written as,

$$P(y = +1 \mid x) / P(y = -1 \mid x) = 1$$

The log-likelihood ratio (LLR) for the above equation can be calculated as,

$$LLR = \log P(y = +1, x) - \log P(y = -1, x)$$

$$= \log\left[\theta_0 \prod_d \theta_{(+1),d}^{[x_d=1]} (1-\theta_{(+1),d})^{[x_d=0]}\right] - \log\left[(1-\theta_0) \prod_d \theta_{(-1),d}^{[x_d=1]} (1-\theta_{(-1),d})^{[x_d=0]}\right]$$

$$= \log\theta_0 - \log(1-\theta_0) + \sum_d [x_d = 1](\log\theta_{(+1),d} - \log\theta_{(-1),d}) + \sum_d [x_d = 0](\log(1-\theta_{(+1),d}) - \log(1-\theta_{(-1),d})$$

The log terms can be simplified as,

$$= \sum_d x_d \log \frac{\theta_{(+1),d}}{\theta_{(-1),d}} + \sum_d (1 - x_d) \log \frac{1 - \theta_{(+1),d}}{1 - \theta_{(-1),d}} + \log \frac{\theta_0}{1 - \theta_0}$$

$$= \sum_d x_d \left[\log \frac{\theta_{(+1),d}}{\theta_{(-1),d}} - \log \frac{1 - \theta_{(+1),d}}{1 - \theta_{(-1),d}} \right] + \sum_d \log \frac{1 - \theta_{(+1),d}}{1 - \theta_{(-1),d}} + \log \frac{\theta_0}{1 - \theta_0}$$

$$= \sum_d x_d \left[\log \frac{(\theta_{(+1),d})(1 - \theta_{(-1),d})}{(\theta_{(-1),d})(1 - \theta_{(+1),d})} \right] + \sum_d \log \frac{1 - \theta_{(+1),d}}{1 - \theta_{(-1),d}} + \log \frac{\theta_0}{1 - \theta_0}$$

Consider, $\log \frac{(\theta_{(+1),d})(1 - \theta_{(-1),d})}{(\theta_{(-1),d})(1 - \theta_{(+1),d})} = w_d$

$$\sum_d \log \frac{1 - \theta_{(+1),d}}{1 - \theta_{(-1),d}} + \log \frac{\theta_0}{1 - \theta_0} = b$$

Therefore, L.L.R = $x.w_d + b$.

3.2 NEURAL NETWORKS

3.2.1 Bio-inspired Multi-layer Networks

Q18. Define neural network. Explain about bio-inspired multi-layer network.

Answer : **Model Paper-II, Q5(b)**

Neural Network

A Neural Network (NN) is a large parallel distributed processor made-up of simple processing 'nodes' or units and connections from one node to another. These units are equivalent to 'neuron' inside a human brain and hence also called as "Neurons" in the network. These are the essential units for processing information throughout the system. The units have tendency for storing previous knowledge and using it further for processing. Each link is directed and is associated with weight from input nodes respectively.

Bio-inspired Multilayer Network

A multilayer network can be defined as a network that contain input layer, output layer and one or more hidden layer. The input layer supply data to the network. The output neurons and the hidden neurons perform the processing of information in the network. The network with hidden neurons acquire global perspective whose function is to extract higher order statistics and provide some connectivity between the input units and output units. The input layer is directed towards the hidden layer which in turn is directed towards its next layer and finally the last layer connects to the output layer which gives the final result. This shows the concept of two or more layers, hence the name multilayer networks.

Example

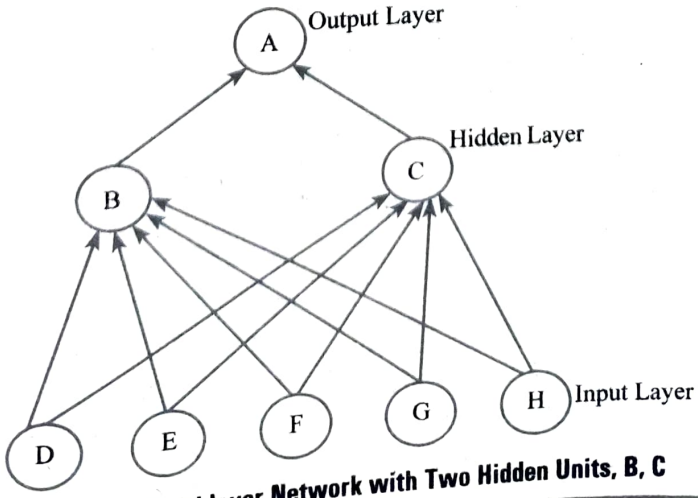

Figure: A Multi-layer Network with Two Hidden Units, B, C

The prediction with neural network is similar to the prediction with a perceptron. This can be done by using below steps,

Step-1: Calculate the activations of the nodes in hidden unit by using the inputs and their weights.

Step-2: Calculate the activation of the nodes in input unit by using the activations calculated in step 1 and the second layer of weights.

The difference between the computation of predication with neural networks and the computation of prediction with perceptron is that, the hidden units in perceptron are calculated as non-linear function of their input units. This function is called as activation function or link function. This can be represented as,

Activation of hidden unit i, $h_i = f(w_i . x)$

Here, f represents link function

w_i represents the vector weight of unit 'i'.

Example

The sign function is considered as an example of link function. It gives the activation as -1 if the incoming signal is negatives, otherwise the activation is considered as $+1$.

Another example of link function is hyperbolic tangent function. The difference between the sign and hyperbolic tangent function is as shown in the below figure,

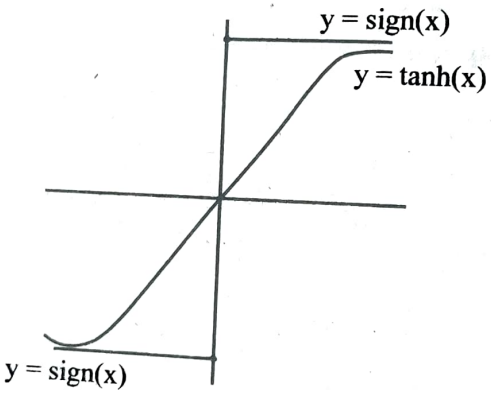

Figure: sign(x) and tanh(x) Functions

As the pictorial representation of sign(x) function is like 'S' which represents a "Sigma" in Greek, this function is also called as "Sigmoid function".

The prediction made by a two-layer network by considering the tanh(x) as link function can be given as.

$$\hat{y} = \sum_i v_i \tanh(w_i . \hat{x})$$

$$\hat{y} = v . \tanh(W\hat{x})$$

Here, W represents the matrix of weights of first layer

v represents the vector of weights of second layer.

Algorithm

The two-layer network prediction algorithm by using hyperbolic tangent function can be given as,

Algorithm (TwoLayerNetworkPredict(W, v, $\hat{x}$))

Inputs : W, v, $\hat{x}$

Outputs : $v.h$

Algorithm :

```
for i = 1 to number-of-hidden-units do
    h_i ← tanh(w_i . x̂)
end for
return v.h
```

3.2.2 The Back-propagation Algorithm, Initialization and Convergence of Neural Networks

Q19. Explain Back Propagation algorithm in detail.

Answer :

Back Propagation

(Model Paper-III, Q15(a) | June/Aug.-22, Q15(b) [OU])

Back propagation network is the most well-known and the widely used technique of teaching Artificial Neural Networks (ANN) about how to do a particular task. It uses supervised learning with more powerful learning rule to implement gradient descent in weight space for a multilayer feed forward network. This learning rule is known as 'back propagation'.

The propagation can be given as,

Back-propagation = Gradient descent + Chain rule

The objective function can be reduced by optimizing the weights in a network. In this, the prediction is non-linear that is $v.\tanh(W\hat{x})$).

The squared error can be optimized by replacing the squared error with the loss function of your own. The main objective is,

$$\min_{W,v} \sum_n \frac{1}{2}\left(y_n \sum_i v_i\, f(w_i.x_n)\right)^2 \quad \ldots (1)$$

Here, f represents a link function similar to tan$h$.

The gradient descent with respect to v can be calculated as,

$$\Delta_v = -\sum_n e_n h_n \quad \ldots (2)$$

Where, e_n denotes the error on the n^{th} sample

h_n denotes the vector of hidden unit activations.

And, the chain rule can be given by,

$$\mathcal{L}(W) = \frac{1}{2}\left(y - \sum_i v_i\, f(w_i.x)\right)^2 \quad \ldots (3)$$

$$\frac{\partial \mathcal{L}}{\partial w_i} = \frac{\partial \mathcal{L}}{\partial f_i}\frac{\partial f_i}{\partial w_i}$$

$$\frac{\partial \mathcal{L}}{\partial f_i} = -\left(y - \sum_i v_i\, f(w_i.x)\right)v_i = -ev_i$$

$$\frac{\partial f_i}{\partial w_i} = f'(w_i.x)x \quad \ldots (4)$$

Solving all the equations, the gradient with respect to w_i can be obtained as,

$$\Delta_{w_i} = -e\, v_i f'(w_i.x)x$$

Algorithm

//Initialize the weights of input layer $W \leftarrow D \times K$ matrix consists of small random values

//Initialize the weights of output layer $v \leftarrow$ K-vector consists of small random values.

for $i = 1$ to MaxIteration do

$G \leftarrow D \times K$(A matrix with zeros)

$g \leftarrow K$(A vector with zeros)

for all $(x, y) \in D$ do

for $i = 1$ to K do

$a_i \leftarrow w_i . \hat{x}$

$h_i \leftarrow \tanh(a_i)$

end for

$\hat{y} \leftarrow v.h$

$e \leftarrow y - \hat{y}$

$g \leftarrow g - eh$

for $i = 1$ to K do

$G_i \leftarrow G_i - ev_i(1 - \tanh^2(a_i))x$

end for

end for

$W \leftarrow W - \eta G$

$v \leftarrow v - \eta g$

end for

return W, v

In the above algorithm, the inputs are (D, η, K, MaxIterations) and outputs are W, v.

Q20. Explain initialization and convergence of neural networks.

Answer :

The initialization of neural network i.e., initializing the values of W (The of weights of first layer) and v (The vector of weights of second layer) to zero creates an uninteresting solution. The direct method of assigning W, v zero leads the model to get stuck in a bad local optimum.

For example, consider the activation (h_i) of the hidden units is zero as $W = 0$. In turn, it says that the gradient on the output weights is 0 in the very first iteration. Thus, weights (v) sets to zero. Eventually, the gradient $w_{1,d}$ d^{th} feature on the first unit will be same as the gradient $w_{2,d}$ for the d^{th} feature on the second unit. There fore, after performing the gradient step, the weight matrix (W) remains same for each hidden unit. Hence, this model converges with a solution which does not use the advantage of accessing the hidden units.

As the initialization of neural network is sensitive, a function called non-convex is used to provide an optimized solution. It consists of plentiful local optima.

Example

Consider the below two-layer network,

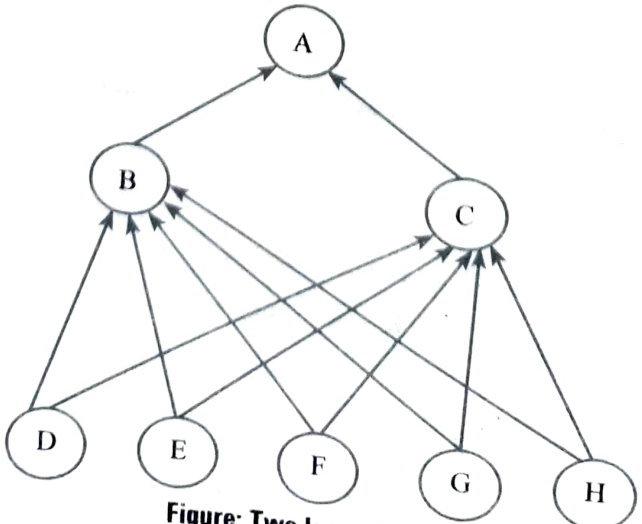

Figure: Two-layer Network

In the above figure, consider the weights from the inputs to the first hidden unit is w_1, to the second hidden unit is w_2, and from the hidden units to the output unit is weights (v_1, v_2). In this, a concept called symmetric modes is introduced that means when the weights are swapped i.e., w_1 with w_2 and v_1 with v_2 the computation will also be same.

Inorder to avoid these problems during the initialization of neural networks, use the random values i.e., −1 to 1 for initialization. So that, there exist less probability of falling into trivial, symmetric local optimum.

Generally, neural networks are considered as difficult, as they have huge count of knobs to tune. They are as follows,

1. The gradient descent learning rate η.
2. The number of layers and hidden units per each layer
3. The initialization
4. The stopping iteration or weight regularization.

Q21. Explain the Back Propagation Algorithm and Initialization and Convergence of Neural Networks.

Answer :

(Model Paper-I, Q15(b) | July/Aug.-22, Q12 [MGU])

Back Propagation Algorithm

For answer refer Unit-III, Page No. 63, Q.No. 19.

Initialization and Convergence of Neural Networks

For answer refer Unit-III, Page No. 64, Q.No. 20.

3.2.3 Beyond Two Layers

Q22. Explain in detail about forward propagation and back propagation algorithms.

Answer :

There are two different versions of algorithms which can go beyond two layers to a arbitrary DAG (arbitrary directed acyclic graph). They are as follows,

1. Forward-propagation algorithm
2. Back-propagation algorithm.

1. Forward-propagation Algorithm

The forward propagation algorithm is used to calculate the activation of the sink y with given inputs. In this algorithm, initially all the activations for the parent nodes of u are calculated and then the activations for the node 'u' will be calculated. This algorithm goes deeper till the child nodes.

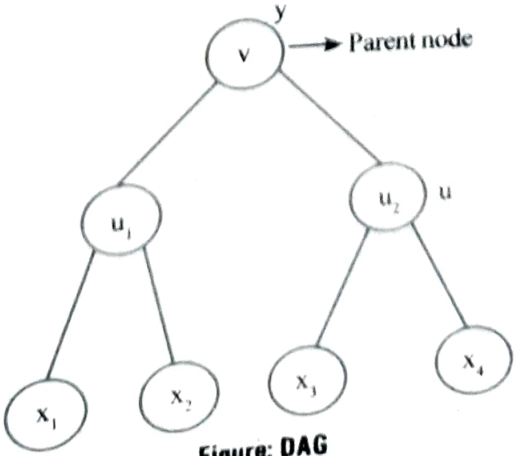

Figure: DAG

In the above figure, consider the weights from the inputs to the first hidden unit is w_1, to the second hidden unit is w_2, and from the hidden units to the output unit is weights (v_1, v_2). In this, a concept called symmetric modes is introduced that means when the weights are swapped i.e., w_1 with w_2 and v_1 with v_2 the computation will also be same.

Inorder to avoid these problems during the initialization of neural networks, use the random values i.e., –1 to 1 for initialization. So that, there exist less probability of falling into trivial, symmetric local optimum.

Generally, neural networks are considered as difficult, as they have huge count of knobs to tune. They are as follows,

1. The gradient descent learning rate η.
2. The number of layers and hidden units per each layer
3. The initialization
4. The stopping iteration or weight regularization.

Q21. Explain the Back Propagation Algorithm and Initialization and Convergence of Neural Networks.

Answer :

(Model Paper-I, Q15(b) | July/Aug.-22, Q12 [MGU])

Back Propagation Algorithm

For answer refer Unit-III, Page No. 63, Q.No. 19.

Initialization and Convergence of Neural Networks

For answer refer Unit-III, Page No. 64, Q.No. 20.

3.2.3 Beyond Two Layers

Q22. Explain in detail about forward propagation and back propagation algorithms.

Answer :

There are two different versions of algorithms which can go beyond two layers to a arbitrary DAG (arbitrary directed acyclic graph). They are as follows,

1. Forward-propagation algorithm
2. Back-propagation algorithm.

1. Forward-propagation Algorithm

The forward propagation algorithm is used to calculate the activation of the sink y with given inputs. In this algorithm, initially all the activations for the parent nodes of u are calculated and then the activations for the node 'u' will be calculated. This algorithm goes deeper till the child nodes.

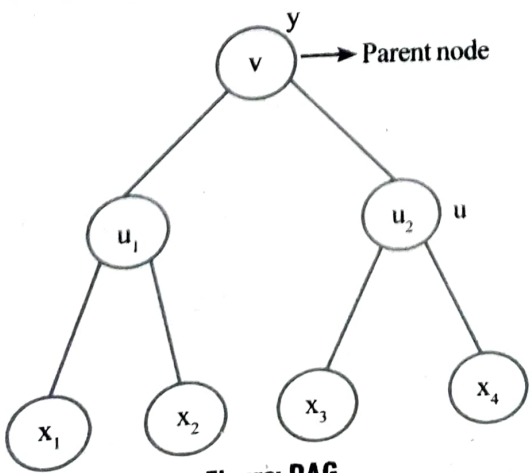

Figure: DAG

Algorithm (ForwardPropagation(x))

for all input nodes u do

$h_u \leftarrow$ corresponding feature of x

end for

for all nodes v in the network do

$a_v \leftarrow \sum_{u \in par(v)} w_{(u,v)} h_u$

$h_v \leftarrow \tanh(a_v)$

end for

return a_y

2. Back-propagation Algorithm

The back-propagation algorithm is used to calculate the derivatives of edge weights for the provided input. The main objective of this algorithm is to compute the errors in backward direction of a network. It also computes the gradients. Here, error refers to the amount of gradient derived from the child nodes.

Algorithm (BackwardPropagation(x, y))

Initially, run the ForwardPropagation(x) //It gives the activations

$e_y \leftarrow y - a_y$

for all nodes v in the network (with error e_v to be computed) do

for all $u \in$ par(v) do

$g_{u,v} \leftarrow -e_v h_u$ //It computes the gradient of all edges

$e_u \leftarrow e_u + e_v w_{u,v}(1 - \tanh^2(a_u))$ //It is the error

end for

end for

return g_e //It returns all gradients.

3.2.4 Breadth Versus Depth, Basis Functions

Q23. Write about deep networks.

Answer :

The deep networks are used in various functions that require only a less or small number of hidden units. **Model Paper-III, Q15(b)(i)**

In some cases, some functions require large number of hidden units when the network is shallow, but requires a small number of hidden units when the network is deep.

Example

Consider an example of parity function that is a generalization of XOR problem. It can be defined as,

$$\text{parity}(x) = \sum_d x_d \bmod 2$$

- → 1 when number of 1s in x is odd
- → 0 when number of 1s in x is even

The parity function can be calculated easily by defining the circuit of depth $O(\log_2 D)$ with $O(D)$ - gates. Here, each gate in XOR is arranged as a complete binary tree.

Parity Function Complexity Theorem

This theorem states that, any circuit of depth $K < \log_2 D$ that calculates the parity function of D input bits should contain Oe^D gates.

The main problem with breadth vs depth is that the number of parameters need to be estimated. On average, each of the parameters needs atleast one example. The number of examples will be less in a deep model than in a shallow model.

Q24. Explain about basis function.

Answer :

Model Paper-III, Q15(b)(ii)

The basis function can be defined as a set of functions in hidden units. It is responsible for creating an arbitrary "basis" for the given vector whenever they are enlarged within the hidden space.

The neural network can be trained by replacing the sigmoid link function by Radial Basis Function (RBF). In RBF, the hidden units are calculated by using the below formula,

$$h_i = \exp[-\gamma_i \| w_i - x\|]^2$$

Where, γ_i represents the width of Gaussian bump

w_i represents the vector weight of unit 'i'.

The RBF network consists of three different layers. They are as follows,

1. **Input Layer**

It consists of various nodes that creates connection between the network and its environment.

2. **Hidden Layer**

It is a single layer a network and performs non-linear transformations from input space to hidden space.

3. **Output Layer**

It consists of various linear nodes that are responsible for sending the results of the network to activation pattern.

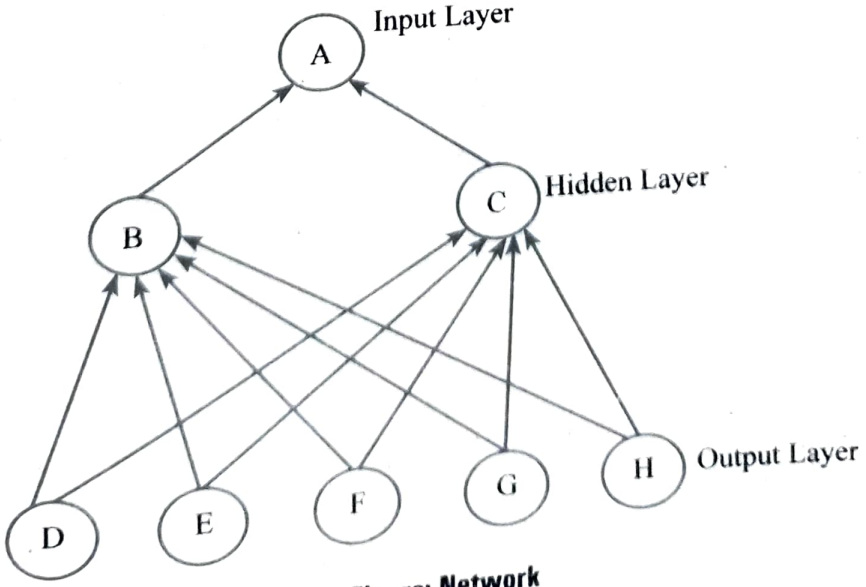

Figure: Network