Deep Learning

Vijaya Saradhi

IIT Guwahati

Mon, 21st Sept 2020

Batch Algorithm

Objective (Cost) Function

- Compute: $\mathbf{w}^T(n)\mathbf{x}(n)$
- Treat the above quantity as the objective function
- With the modification $\mathbf{w}^T(n)\mathbf{x}(n)d(n)$
- For one $\mathbf{x}(n)$ the above objective function is used:
- For many $\mathbf{x}(n)$'s we have:

$$J(\mathbf{w}) = \sum_{\mathbf{x}(n)\in\mathcal{H}} \left(-(\mathbf{w}^T(n)\mathbf{x}(n)d(n))\right)$$

- The above objective function should be minimized

Batch Algorithm

Objective Function - Intuition

- We have to minimize or maximize a given objective function
- Percentron rule: $\mathbf{w}^T(n)\mathbf{x}(n) > 0$ $\mathbf{x}(n) \in \mathcal{C}_1$
- $\mathbf{w}^T(n)\mathbf{x}(n)$ is negative for $\mathcal{C}_2$, d(n) = -1. Add all these terms.
- $\mathbf{w}^T(n)\mathbf{x}(n)$ is positive for $\mathcal{C}_1$ and d(n) $= +1$. Decrease it by multiplying it -1
- Percentron rule: $\mathbf{w}^T(n)\mathbf{x}(n) \leq 0$ $\mathbf{x}(n) \in \mathcal{C}_1$
- That is for any $\mathbf{x}(n)$, the quantity $-(\mathbf{w}^T(n)\mathbf{x}(n)d(n))$ to be minimized

Batch Algorithm

Apply Gradient Descent Rule

- Compute direction: $\bigtriangledown J(\mathbf{w}) = \sum_{\mathbf{x}(n)\in\mathcal{H}} (-\mathbf{x}(n)d(n))$
- Update $\mathbf{w}(n+1) = \mathbf{w}(n) - \eta(n) \bigtriangledown J(\mathbf{w})$
- That is $\mathbf{w}(n+1) = \mathbf{w}(n) - \eta(n) \sum_{\mathbf{x}(n)\in\mathcal{H}} (-\mathbf{x}(n)d(n)))$

Multi-layer Perceptrons

Perceptron

- Perceptrons works for linearly separable data
- Activation function is threshold function
- Not differentiable one $sgn(\mathbf{w}^T\mathbf{x})$ w.r.t. $\mathbf{w}$

Multi-layer Perceptron

- Each neuron includes a non-linear activation function that is differentiable
- Network contains one or more hidden layers
- Each neuron is connected with every other neuron to its immediate next layer

Multi-layer Perceptrons

Training

Forward phase Weights are fixed; input signal is propagated through the network layer by layer until it reaches the output. Changes are observed only in activation potentials and outputs

Backword phase Significantly different

- Error is computed
- The error is propagated backwards to adjust the weights of the network

MLP example network

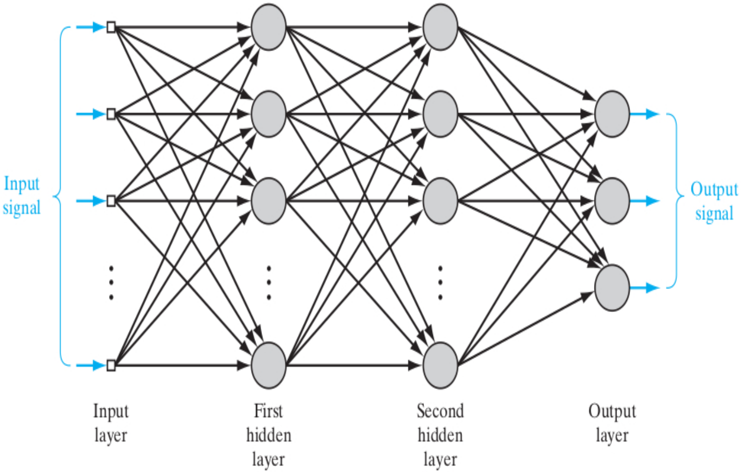

FIGURE 4.1 Architectural graph of a multilayer perceptron with two hidden layers.

Error Propagation

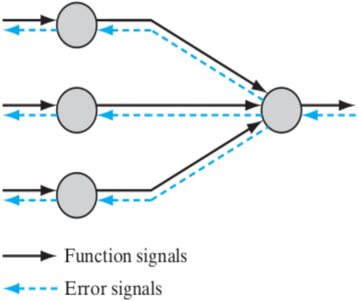

FIGURE 4.2 Illustration of the directions of two basic signal flows in a multilayer perceptron: forward propagation of function signals and back propagation of error signals.

Multi-layer Perceptrons

Types of Signals

Function Signal characterized by forward direction of information flow

- An input (signal) originates at input layer.
- Propagates forward neuron by neuron.
- Reaches output layer and produces the output

Multi-layer Perceptrons

Types of Signals

Error Signal characterized by backward direction information flow

- Compute the error produced at an output neuron
- Propagate the error backward layer by layer
- Weight adjustment depends on the error that is propagated backwards as well

Multi-layer Perceptrons

Multiple Outputs

- So far the output is of the form {spam, not-a-spam}, {0, 1}, {+1, -1}
- Output belongs to a set of class labels such as $\{0,\ 1,\ \cdots,\ 9\}$
- Activation functions
  - Threshold {0, 1}
  - Signum function {+1, -1}
  - Sigmoid $\frac{1}{(1+exp(-av))}$ output: $[0, 1]$

Multi-layer Perceptrons

Multiple Outputs

- Have as many bits in the output as there are number of classes
- Represent them using one-hot encoding
- That is only one bit can take value 1 and all other must take value 0

Multi-layer Perceptrons

Example

Class labels {'setosa', 'versicolor', 'virginica'}: 3
Number of output bits: 3

$$\textit{setosa} \quad \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{1}$$

Multi-layer Perceptrons

Example

Class labels {'setosa', 'versicolor', 'virginica'}: 3
Number of output bits: 3

$$\textit{versicolor} \quad \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \tag{2}$$

Multi-layer Perceptrons

Example

Class labels {'setosa', 'versicolor', 'virginica'}: 3
Number of output bits: 3

$$\textit{virginica} \quad \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{3}$$

Multi-layer Perceptrons

Multiple Outputs

Number of neurons in output layer is given by number of distinct classes in training dataset

Multi-layer Perceptrons

Error function

- Of the form $\{\mathbf{x}(n), \mathbf{d}(n)\}_{n=1}^{N}$
- Let $y_j(n)$: function signal at the output neuron j in the output layer
- Error at neuron j is: $e_j(n) = d_j(n) - y_j(n)$
- Instantaneous error energy:

$$\mathcal{E}_j(n) = \frac{1}{2} e_j^2(n)$$

- The above equation is error made by one output neuron

Multi-layer Perceptrons

Error function

- Total instantaneous error energy is:

$$\begin{aligned} \mathcal{E}(n) &= \sum_{j \in C} \mathcal{E}_j(n) \\ &= \frac{1}{2} e_j^2(n) \end{aligned}$$

- The above equation is for one training example.
- Error incurred over all the training examples is given by:
- Average error

$$\begin{aligned} \mathcal{E}_{av}(N) &= \frac{1}{N} \sum_{n=1}^{N} \mathcal{E}(n) \\ &= \frac{1}{2N} \sum_{n=1}^{N} \sum_{j \in C} e_j^2(n) \end{aligned}$$