

1. Linear neural networks

- (a) Suppose we have a three layer neural network with one input layer x , one hidden layer h , and one output layer y . Each layer can be expressed as a vector of the values of the nodes in that layer. For example,

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \quad (1)$$

Assume that each neural node has its own set of weights \mathbf{w}_i where i is the node index. We can express the value of a particular output in terms of the hidden layer

$$y_k = c_{y_k} \cdot \mathbf{w}_{\mathbf{y}_k} \cdot \mathbf{h} \quad (2)$$

where c_{y_k} is the constant multiplier of y_k , w_{y_k} is the set of weights for y_k , and h is the vector of hidden nodes.

Similarly, we can express the value of each node in the hidden layer in terms of the inputs.

$$h_j = c_{h_j} \cdot \mathbf{w}_{\mathbf{h}_j} \cdot \mathbf{x} \quad (3)$$

Now, we can see that the output layer nodes can simply be written in terms of the inputs without the hidden layer. For a particular output

node:

$$\begin{aligned}
y_k &= c_{y_k} \cdot \mathbf{w}_{y_k} \cdot \mathbf{h} \\
&= c_{y_k} \cdot \mathbf{w}_{y_k} \cdot \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_n \end{pmatrix} \\
&= c_{y_k} \cdot \mathbf{w}_{y_k} \cdot \begin{pmatrix} c_{h_1} \cdot \mathbf{w}_{h_1} \cdot \mathbf{x} \\ c_{h_2} \cdot \mathbf{w}_{h_2} \cdot \mathbf{x} \\ \vdots \\ c_{h_n} \cdot \mathbf{w}_{h_n} \cdot \mathbf{x} \end{pmatrix} \\
&= c_{y_k} \cdot \begin{pmatrix} w_{y_k 1} & w_{y_k 2} & \dots & w_{y_k n} \end{pmatrix} \begin{pmatrix} c_{h_1} \cdot \mathbf{w}_{h_1} \cdot \mathbf{x} \\ c_{h_2} \cdot \mathbf{w}_{h_2} \cdot \mathbf{x} \\ \vdots \\ c_{h_n} \cdot \mathbf{w}_{h_n} \cdot \mathbf{x} \end{pmatrix} \\
&= c_{y_k} \sum_{i=1}^n w_{y_k i} \cdot c_{h_i} \cdot \mathbf{w}_{h_i} \cdot \mathbf{x} \\
&= c_{y_k} \left(\sum_{i=1}^n w_{y_k i} \cdot c_{h_i} \cdot \mathbf{w}_{h_i} \right) \mathbf{x}
\end{aligned}$$

- (b) For an arbitrary number of hidden nodes, the same computation can be done. We demonstrate below with two hidden layers: h_m, h_n
- (c) For the case when $h \ll n$, a neural net with the hidden layer will do $O(hn)$ computations to find the linear combination of the weighted sum of inputs whereas without the hidden layer, as shown in (a), the output is only dependent on x . This computation is $O(n)$, so we save those $h - 1$ other computations over the inputs.

2. ML estimation of exponential model

Knowing

$$P(x) = \frac{1}{b} e^{-\frac{x}{b}}$$

- (a) We write the likelihood function given x_i as

$$\begin{aligned}
\mathcal{L}(b|x_1, \dots, x_N) &= \prod_{i=1}^N \frac{1}{b} e^{-\frac{x_i}{b}} \\
&= \left(\frac{1}{b}\right)^N \prod_{i=1}^N e^{-\frac{x_i}{b}} \\
&= \left(\frac{1}{b}\right)^N e^{\sum_{i=1}^N -\frac{x_i}{b}}
\end{aligned}$$

(b) We first find

$$\begin{aligned}
\log(\mathcal{L}) &= \log \left(\left(\frac{1}{b} \right)^N e^{\sum_{i=0}^N \frac{x_i}{b}} \right) \\
&= \log \left(\left(\frac{1}{b} \right)^N \right) + \log \left(e^{\sum_{i=0}^N \frac{x_i}{b}} \right) \\
&= n(\log(1) - \log(b)) + \sum_{i=0}^N \frac{x_i}{b} \log(e)
\end{aligned}$$

Then,

$$\begin{aligned}
\frac{\partial \log \mathcal{L}}{\partial b} &= \frac{\partial N(\log(1) - \log(b))}{\partial b} + \frac{\partial \sum_{i=0}^N \frac{x_i}{b} \log(e)}{\partial b} \\
&= -\frac{N}{b} + \frac{\partial \sum_{i=0}^N x_i \cdot \log(e)}{\partial b} \\
&= -\frac{N}{b} - \frac{N}{b^2} \sum_{i=0}^N x_i \cdot \log(e) \\
&= -\frac{N}{b} \left(1 - \frac{1}{b} \sum_{i=0}^N x_i \log(e) \right)
\end{aligned}$$

(c) We aim to maximize \mathcal{L} so,

$$\frac{\partial \mathcal{L}}{\partial b} = -\frac{N}{b} \left(1 - \frac{1}{b} \sum_{i=0}^N x_i \log(e) \right) = 0$$

We can reassemble this as

$$\begin{aligned}
-\frac{N}{b} \left(1 - \frac{1}{b} \log(e) \sum_{i=0}^N x_i \right) &= 0 \\
-N + \frac{N}{b} \log(e) \sum_{i=0}^N x_i &= 0 \\
\frac{N}{b} \log(e) \sum_{i=0}^N x_i &= N \\
N \log(e) \sum_{i=0}^N x_i &= Nb \\
\log(e) \sum_{i=0}^N x_i &= b
\end{aligned}$$

3. ML estimation of noisy-OR model