CMSC35000-1 Introduction to Artificial Intelligence **Winter 2005**

Lecture 4: January 18

Lecturer: Partha Niyogi *Scribe: Henry Wu*

4.0.0.1 Topics: Multilayer Perceptrons

4.0.0.2 1. Review

· We use perceptrons to simulate learning.

· In an ideal world, we know the distribution P on $X \times Y$, we have an optimal classifier, Bayes discriminant function. This is $g : X \to Y$ s.t. $g(x) = 1 \Leftrightarrow P(Y = 1|x) > \frac{1}{2}$. g is the best possible classifier, naturally defined.

· Normally, we don't have g . We can only derived $\widehat{g}$ from some random examples $\{(x_i, y_i)\}$. In the case of preceptrons, $\widehat{g}$ is of the form $\theta(w \cdot x)$.

· The limitation of perceptrons: Data need to be separable, otherwise a perceptron does not exist.

4.0.0.3 2. Multilayer Perceptrons

· Above discussion motivates us to consider Multilayer Perceptrons (MLPs) , also called neural networks.

· Our convention: $x \in R^n$, $y \in \{0, 1\}$

· What we originally have is one neuron, now we have many neurons:

	○			output layer
	...			
○	○	...	○	layer 2 (hidden layer)
○	○	...	○	layer 1 (hidden layer)
x_1	x_2	...	x_n	input layer

· This is also called a "feed forward architecture" for there is no feed back.

· The whole network computes non-linear functions. This looks more like our brain.

· It was unknown how to train this network until 1986. A good reference about this is "Perceptrons" by Minsky and Papert, 1969.

4.0.0.4 3 Backpropagation Algorithm

· In 1986, Rumelhart, Hinton, and McClelland proposed Backpropagation Algorithm to compute and train neural networks.

· Notation:

$O_i^{(k)}$: The output of the i^{th} perceptron of the k^{th} layer.

K : The number of layers. ($O^{(K)}$ is the output node.)

$w_{ij}^{(k)}$: The link of node i in the $(k-1)^{th}$ layer and node j in the k^{th} layer.

n_k : The number of nodes in the k^{th} layer. ($n_K = 1$)

$(x_i, y_i) : i = 1, 2, 3, ..., l$ are our original data. $x_i \in R^N$. $y_i \in \{0, 1\}$.

· We define $\theta(z) = 1$ when $z > 0$, 0 otherwise. We have $O_i^{(k)} = \theta(\sum_{j=1}^{n_{k-1}} w_{ji}^{(k)} O_j^{(k-1)})$.

· We define $J(\{w_{ij}^{(k)}\}) = \sum_{t=1}^{l} (y_t - O^{(K)}(x_t, \{w_{ij}^{(k)}\}))^2$. We want to minimize J . From the form we may see the intrinsic difficulty.

· The inovation of the paper is using the method of Gradient Descent. We have a function J and want to find the minimum. We may compute its gradient. ¿From the k^{th} iterative data Z_k we may compute the gradient $\nabla_Z J|_{Z=Z_k}$ and let $Z_{k+1} = Z_k - \varepsilon \nabla_Z J|_{Z=Z_k}$. When the gradient is 0 , we will be at the local minimum.

· Two problems: (We won't discuss here.)

1. How to deal with local minimum v.s. global minimum? i.e. How to make sure that we may find the global minimum which is we want?

2. (Numerical problem) How to avoid overshooting and make our solution sequence converge?

· To solve the problem that J is not differentiable because θ is a step function, we use the sigmoidal function $\sigma(z) = \frac{1}{1+e^{-\alpha z}}$, where $\alpha > 0$.

· The properties of $\sigma(z)$:

1. $0 < \sigma(z) < 1$

2. $\sigma(z) \to 1$ when $z >> 0$

3. $\sigma(z) \to 0$ when $z << 0$

4.0.0.5 4. The Method of Greedy Descent and The Computation

· We use the method of Greedy Descent to solve our problem.

· Let W_i be weight vectors $\begin{pmatrix} w_{ij}^{(k)} \\ \ldots \end{pmatrix}$. We have the formula $W_{L+1} = W_L - \varepsilon_L \nabla_W J|_{W=W_L}$

. To compute $\nabla_W J$, we need to compute $\frac{\partial J}{\partial w_{ij}^{(k)}}$. By the chain rule,

$$\frac{\partial J}{\partial w_{ij}^{(k)}} = \sum_{m=1}^{n_k} \frac{\partial J}{\partial O_m^{(k)}} \cdot \frac{\partial O_m^{(k)}}{\partial w_{ij}^{(k)}}$$

$$= \frac{\partial J}{\partial O_j^{(k)}} \cdot \frac{\partial O_j^{(k)}}{\partial w_{ij}^{(k)}} \text{ (Because only } O_j^{(k)} \text{ depends on } w_{ij}^{(k)} \text{ .)}$$

So now we want to compute $\frac{\partial J}{\partial O_j^{(k)}}$ and $\frac{\partial O_j^{(k)}}{\partial w_{ij}^{(k)}}$.

· To compute $\frac{\partial O_j^{(k)}}{\partial w_{ij}^{(k)}}$, because $O_j^{(k)} = \sigma(\sum_{m=1}^{n_{k-1}} w_{mj}^{(k)} O_m^{(k-1)})$, $\frac{\partial O_j^{(k)}}{\partial w_{ij}^{(k)}} = \frac{d\sigma}{dz} \cdot \frac{\partial z}{\partial w_{ij}^{(k)}}$ where

$z = \sum_{m=1}^{n_{k-1}} w_{mj}^{(k)} O_m^{(k-1)}$.

· For $\sigma(z) = \frac{1}{1+e^{-\alpha z}}$, $\frac{d\sigma}{dz} = -\frac{1}{(1+e^{-\alpha z})^2} \cdot e^{-\alpha z} \cdot (-\alpha) = \frac{\alpha e^{-\alpha z}}{(1+e^{-\alpha z})^2} = \alpha \cdot \frac{1}{1+e^{-\alpha z}} \cdot \frac{e^{-\alpha z}}{1+e^{-\alpha z}}$

$= \alpha\sigma(z)(1-\sigma(z))$.

· Hence we have $\frac{\partial O_j^{(k)}}{\partial w_{ij}^{(k)}} = \frac{d\sigma}{dz} \cdot \frac{\partial z}{\partial w_{ij}^{(k)}} = \alpha O_j^{(k)}(1-O_j^{(k)})O_i^{(k-1)}$

· To compute $\frac{\partial J}{\partial O_j^{(k)}}$, by chain rule, $\frac{\partial J}{\partial O_j^{(k)}} = \sum_{m=1}^{n_{k+1}} \frac{\partial J}{\partial O_m^{(k+1)}} \cdot \frac{\partial O_m^{(k+1)}}{\partial O_j^{(k)}}$. Because

$O_m^{(k+1)} = \sigma(\sum_{n=1}^{n_k} w_{nm}^{k+1} O_n^{(k)})$, we have $\frac{\partial O_m^{(k+1)}}{\partial O_j^{(k)}} = \frac{d\sigma}{dz} \cdot \frac{\partial z}{\partial O_j^{(k)}}$ (where $z = \sum_{n=1}^{n_k} w_{nm}^{k+1} O_n^{(k)}$)

$= \alpha O_m^{(k+1)}(1-O_m^{(k+1)})w_{jm}^{(k+1)}$. We still have to compute $\frac{\partial J}{\partial O_m^{(k+1)}}$.

· The above results yield a method to compute all $\frac{\partial J}{\partial w_{ij}^{(k)}}$. We first compute $\frac{\partial J}{\partial O^{(K)}}$, then

$\frac{\partial J}{\partial O_i^{(K-1)}}, \frac{\partial J}{\partial O_i^{(K-2)}}, \ldots$, and we can have all derivatives. It is the reason that why we call it

Backpropagation. From $J = \sum_{t=1}^{l}(y_t - O^{(K)})^2$ we have $\frac{\partial J}{\partial O^{(K)}} = -\sum_{t=1}^{l} 2(y_t - O^{(K)})$. Hence

we are able to compute by the relations we have.

4.0.0.6 5. Conclusion

· The above is the end of the theory of neural networks.

· The key problem for multilayer perceptrons: The choice of the topology of the structure.

· In practice, people use one hidden layer with many nodes (200˜300 , for example) .