ELEG 5040: Homework #1

Xingyu ZENG

March 7, 2015

1 Problem 1

Suppose three layers to be x, h, y. See Figure 1.

$$h_1 = \sigma(x_1 + 1) \tag{1}$$
$$h_2 = \sigma(x_2 + 1) \tag{2}$$
$$h_3 = \sigma(1 - x_1 - 2x_2) \tag{3}$$
$$y_1 = \sigma(2.5 - h_1 - h_2 - h_3) \tag{4}$$
$$y_2 = \sigma(h_1 + h_2 + h_3 - 2.5) \tag{5}$$

Here, $\sigma(x)$=1 if $x \geq 0$ else $\sigma(x)$=0;

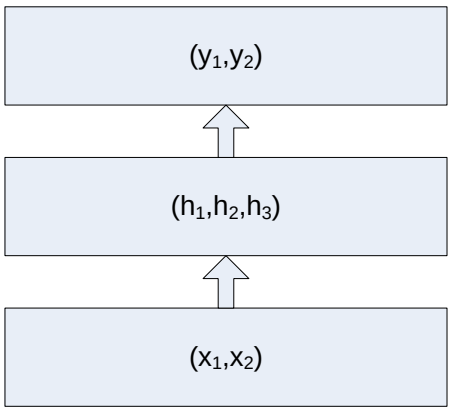

Figure 1: Three layer network

2 Problem 2

The network has 3 layer, the first layer has the same number dimension as the input x, the second layer has n nodes which is the number of support vectors, and the last layer has only node which outputs 1 if the input data x satisfy the svm conditions.

Let the second layer to be h=[h_1,h_2...h_n], the last layer to be y.

$$h_i = \sum_{j=1}^{d} |x_j - x_{i,j}|^2 \tag{6}$$

$$\bar{h}_i = exp(-\frac{h_i}{\sigma^2}) \tag{7}$$

$$y = \sum_{i=1}^{n} \alpha_i \bar{h}_i \tag{8}$$

$$\bar{y} = \sigma(y) \tag{9}$$

where $\sigma(y)$=1 if y $\geq$ 0 else $\sigma(y)$=0

3 Problem 3

$$f(x)[i,j] = \sum_m \sum_n w[m,n]x[i-m, j-n] \tag{10}$$

$$g(x)[i,j] = x[i-u, j-v] \tag{11}$$

$$\Rightarrow f(g(x))[i,j] = \sum_m \sum_n w[m,n]g(x)[i-m, j-n] \tag{12}$$

$$= \sum_m \sum_n w[m,n]x[i-m-u, j-n-v] \tag{13}$$

$$= \sum_m \sum_n w[m,n]x[(i-u)-m, (j-v)-n] \tag{14}$$

$$= f(x)[i-u, j-v] = g(f(x)) \tag{15}$$

Thus convolution has equivariance to translation.

$$f(x)[i,j] = \sum_m \sum_n w[m,n]x[i-m, j-n] \tag{16}$$

$$s(x)[i,j] = x[u*i, v*j] \tag{17}$$

$$\Rightarrow f(s(x))[i,j] = \sum_m \sum_n w[m,n]s[i-m, j-n] \tag{18}$$

$$= \sum_m \sum_n w[m,n]s[u*(i-m), v*(j-n)] \tag{19}$$

$$\neq \sum_m \sum_n w[m,n]s[u*i-m, v*j-n] \tag{20}$$

$$= f(x)[u*i, v*j] = s(f(x)) \tag{21}$$

Thus Convolution is not equivariant to downsampling.

4 Problem 4

The output of k-stride convolution can be regarded as the output of one k-step pooling applied on the output of 1-stride convolution.

When the convolution stride is k,

$$net_j = (x \otimes w)[j * k] = \sum_u w_u * x_{j*k-u} \tag{22}$$

$$\Rightarrow \frac{\partial L}{\partial w_m} = \sum_j \frac{\partial L}{\partial net_j} * \frac{\partial net_j}{\partial w_m} = -\sum_j \delta_j x_{j*k-m} \tag{23}$$

$$\tag{24}$$

Here $\delta_j = -\frac{\partial L}{\partial net_j}$ is the j-th element of sensitivity map.

For 2D case,

$$net_{i,j} = (x \otimes w)[i * k, j * k] = \sum_{u,v} w_{u,v} * x_{i*k-u,j*k-v} \tag{25}$$

$$\Rightarrow \frac{\partial L}{\partial w_{m,n}} = \sum_{i,j} \frac{\partial L}{\partial net_{i,j}} * \frac{\partial net_{i,j}}{\partial w_{m,n}} = -\sum_{i,j} \delta_{i,j} x_{i*k-m,j*k-n} \tag{26}$$

Here $\delta_{i,j} = -\frac{\partial L}{\partial net_{i,j}}$ is the i,j-th element of sensitivity map.

5 Problem 5

5.1 Squared Error Case

$$\frac{\partial z_k}{\partial net_k} = \frac{e^{net_k}}{(1+e^{-net_k})^2} = \frac{1}{1+e^{-net_k}} - \frac{1}{(1+e^{-net_k})^2} = z_k * (1 - z_k) \tag{27}$$

$$\Rightarrow \delta_k = -\frac{\partial J}{\partial net_k} = -\frac{\partial J}{\partial z_k} * \frac{\partial z_k}{\partial net_k} \tag{28}$$

$$= -(t_k - z_k) * \frac{\partial z_k}{\partial net_k} = (z_k - t_k) * z_k * (1 - z_k) \tag{29}$$

When z_k is near to 0 or 1, δ_k will be close to 0 even if $z_k - t_k$ is large. The network will not update parameters if such situation happens and the prediction will remain far away from the target.

5.2 Cross Entropy Case

$$\frac{\partial z_k}{\partial net_k} = \frac{e^{net_k} * \sum_{k'} e^{net_{k'}} - e^{net_k} * e^{net_k}}{(\sum_{k'} e^{net_{k'}})^2} = z_k(1 - z_k) \tag{30}$$

$$\frac{\partial z_{k''}}{\partial net_k} = \frac{-e^{net_k} * e^{net_k''}}{(\sum_{k'} e^{net_{k'}})^2} = -z_k * z_{k''} \tag{31}$$

$$\Rightarrow \delta_k = -\frac{\partial J}{\partial net_k} = -\sum_{k'} \frac{\partial J}{\partial z_{k'}} * \frac{\partial z_{k'}}{\partial net_k} \tag{32}$$

$$= \sum_{k'} \frac{t_{k'}}{z_{k'}*} * \frac{\partial z_{k'}}{\partial net_k} = t_k(1 - z_k) - \sum_{k'' \neq k} t_{k''} * z_k \tag{33}$$

$$= t_k - t_k * z_k - z_k * (1 - t_k) = t_k - z_k \tag{34}$$

Because the nonlinear activation function is softmax, the prediction sum is to be 1. If the prediction error is large, then for the element with target value to be 1, the prediction value will be far away from 1. That δ_k will be large.