

word2vec

Kristian Wichmann

May 2, 2017

1 A simple neural network

Consider a feed forward neural network with one hidden layer, as shown in figure 1. The input layer consists of a row¹ vector x of dimension D_x , i.e. $x \in \mathbb{R}^{1 \times D_x}$. The hidden layer has h neurons with a sigmoid activation function:

$$h = \sigma(xW^{(1)} + b^{(1)}) = \sigma(z^{(1)}), \quad z^{(1)} = xW^{(1)} + b^{(1)} \quad (1.1)$$

So $h \in \mathbb{R}^{1 \times H}$, $W^{(1)} \in \mathbb{R}^{D_x \times H}$, $b^{(1)} \in \mathbb{R}^{1 \times H}$. The output layer has D_y softmax neurons:

$$\hat{y} = s(hW^{(2)} + b^{(2)}) = s(z^{(2)}), \quad z^{(2)} = hW^{(2)} + b^{(2)} \quad (1.2)$$

Similarly $\hat{y} \in \mathbb{R}^{1 \times D_y}$, $W^{(2)} \in \mathbb{R}^{H \times D_y}$, $b^{(2)} \in \mathbb{R}^{1 \times D_y}$.

¹Note that this is different from the column vector convention usually used.

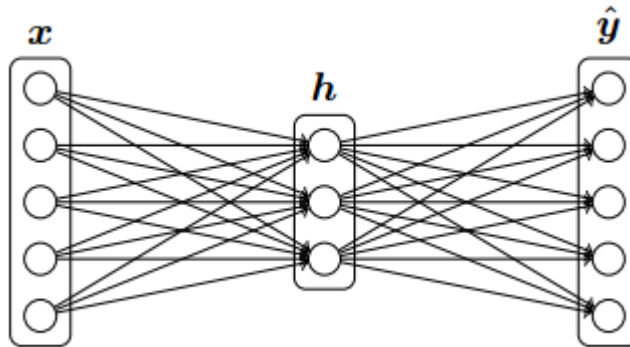


Figure 1: The neural network.

1.1 Error function

Assuming a labelled dataset x with the correct label c encoded as a one-hot vector $t \in \mathbb{R}^{1 \times D_y}$, so $t_i = \delta_{ic}$. We will use the cross-entropy error function:

$$J(x) = - \sum_{i=1} t_i \log \hat{y}_i \quad (1.3)$$

Since t is one-hot encoded, only the correct label c will contribute to the sum, so:

$$J(x) = - \log \hat{y}_c \quad (1.4)$$

This does not mean that the other components of \hat{y} will not matter, since the softmax indirectly depends on all components.

1.1.1 Derivative with respect to input

We now wish to compute the derivative of $J(x)$ with respect to x . In shorthand, the chain rule gives us:

$$\frac{\partial J}{\partial x} = \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z^{(2)}} \frac{\partial z^{(2)}}{\partial h} \frac{\partial h}{\partial z^{(1)}} \frac{\partial z^{(1)}}{\partial x} \quad (1.5)$$

Writing out indices explicitly:

$$\frac{\partial J}{\partial x_m} = \sum_{i=1}^{D_y} \sum_{j=1}^{D_y} \sum_{k=1}^H \sum_{l=1}^H \frac{\partial J}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial z_j^{(2)}} \frac{\partial z_j^{(2)}}{\partial h_k} \frac{\partial h_k}{\partial z_l^{(1)}} \frac{\partial z_l^{(1)}}{\partial x_m} \quad (1.6)$$

Let's compute these partial derivatives one by one:

$$\frac{\partial J}{\partial \hat{y}_i} = - \frac{\partial}{\partial \hat{y}_i} \log \hat{y}_c = - \frac{\delta_{ic}}{\hat{y}_c} \quad (1.7)$$

The second is a standard result for the softmax function:

$$\frac{\partial \hat{y}_i}{\partial z_j^{(2)}} = s_i(z^{(2)}) (\delta_{ij} - s_j(z^{(2)})) = \hat{y}_i (\delta_{ij} - \hat{y}_j) \quad (1.8)$$

And:

$$\frac{\partial z_j^{(2)}}{\partial h_k} = W_{kj}^{(2)} \quad (1.9)$$

The fourth uses a standard result for the sigmoid:

$$\frac{\partial h_k}{\partial z_l^{(1)}} = \delta_{kl} \sigma(z_l^{(1)}) (1 - \sigma(z_l^{(1)})) = \delta_{kl} h_l (1 - h_l) \quad (1.10)$$

And finally:

$$\frac{\partial z_l^{(1)}}{\partial x_m} = W_{ml}^{(1)} \quad (1.11)$$

Inserting, letting the delta functions cancel, and renaming indices, this becomes:

$$\frac{\partial J}{\partial x_i} = - \sum_{j=1}^H \sum_{k=1}^{D_y} W_{ij}^{(1)} h_j (1 - h_j) W_{jk}^{(2)} (\delta_{kc} - \hat{y}_k) \quad (1.12)$$

But since $t_i = \delta_{ic}$ this is the same as:

$$\frac{\partial J}{\partial x_i} = \sum_{j=1}^H \sum_{k=1}^{D_y} W_{ij}^{(1)} h_j (1 - h_j) W_{jk}^{(2)} (\hat{y}_k - t_k) \quad (1.13)$$

We may rephrase this in terms of *errors* for the different layers. The error in the output layer is:

$$\delta_k^{(2)} = \hat{y}_k - t_k \quad (1.14)$$

And the backpropagated error in the hidden layer is:

$$\delta_j^{(1)} = \sum_{k=1}^{D_y} h_j (1 - h_j) W_{jk}^{(2)} \delta_k^{(2)} \quad (1.15)$$

Now equation 1.13 can be rewritten as:

$$\frac{\partial J}{\partial x_i} = \sum_{j=1}^H \sum_{k=1}^{D_y} W_{ij}^{(1)} h_j (1 - h_j) W_{jk}^{(2)} \delta_k^{(2)} = \sum_{j=1}^H W_{ij}^{(1)} \delta_j^{(1)} \quad (1.16)$$

2 The word2vec algorithm