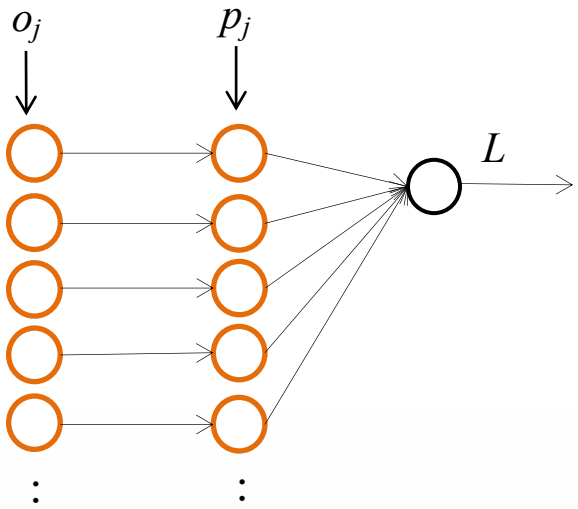
Notes on Homework 1

1. There will be bias neurons at each layer. Biases can be initialized as 1. They will be updated at each batch like the standard neurons.

2. There will not be any activation function (ReLU) in the output layer.

3. Your loss is computed by softmax function. Therefore, loss in the output layer, $\delta^{(3)}$, will be computed by the derivative of loss function.



Softmax function:

$$p_j = \frac{e^{o_j}}{\sum_k e^{o_k}}$$

where o_j are the output neurons (scores). This is used in the softmax function of the form:

$$L = -\sum_j y_j \log p_j,$$

In MNIST data, loss consists of only one character (correct label). But in general, softmax loss contains softmax function of all output neurons.

Derivative of softmax function w.r.t. the output neurons is as follows
(for details, see https://deepnotes.io/softmax-crossentropy):

$$\frac{\partial p_j}{\partial o_i} = p_j(1 - p_i), \quad i = j$$

$$\frac{\partial p_j}{\partial o_i} = -p_i p_j, \quad i \neq j.$$

Then, the derivative of the loss on output neurons can be obtained as (k is used instead of j):

$$\frac{\partial L}{\partial o_i} = -\sum_k y_k \frac{\partial \log p_k}{\partial o_i} = -\sum_k y_k \frac{1}{p_k} \frac{\partial p_k}{\partial o_i}$$

$$= -y_i(1 - p_i) - \sum_{k \neq i} y_k \frac{1}{p_k}(-p_k p_i)$$

$$= -y_i(1 - p_i) + \sum_{k \neq i} y_k (p_i)$$

$$= -y_i + y_i p_i + \sum_{k \neq i} y_k (p_i)$$

$$= p_i \left(\sum_k y_k \right) - y_i = p_i - y_i$$

So, it turns out to be that the derivative becomes (output-target) for each neuron.
You can assign your $\delta^{(3)}$ as output - target.

(source: https://math.stackexchange.com/questions/945871/derivative-of-softmax-loss-function)