HW-3：Neural Network

Deadlines: **2016.05.10**-23:59:59

In this homework, you are asked to build **neural network** models for the classification problem of the **hand-written numbers** $\boldsymbol{x}$. Before doing so, the following layers need to be implemented as a Matlab function respectively:

- **Inner-product layer**
- **Activation layer**
- **Softmax layer**

Besides, it is required to use different types of the **activation function** for the tasks:

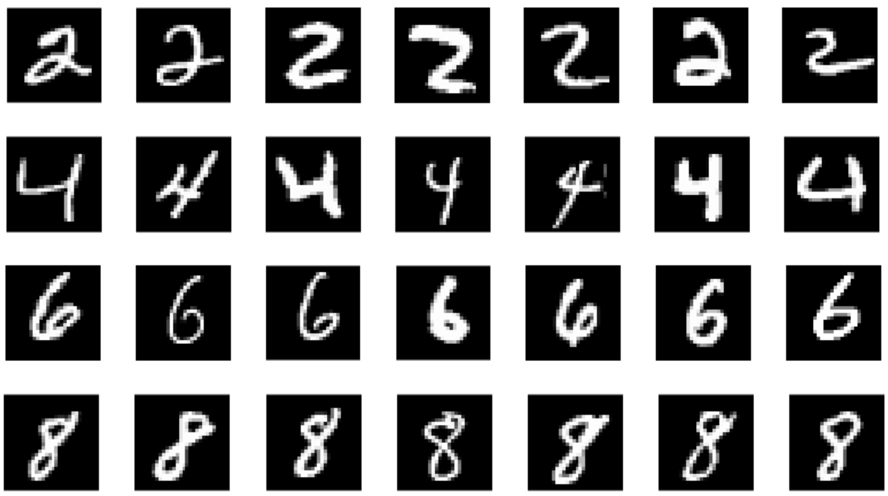
- **Sigmoid function**
- **Rectified function**

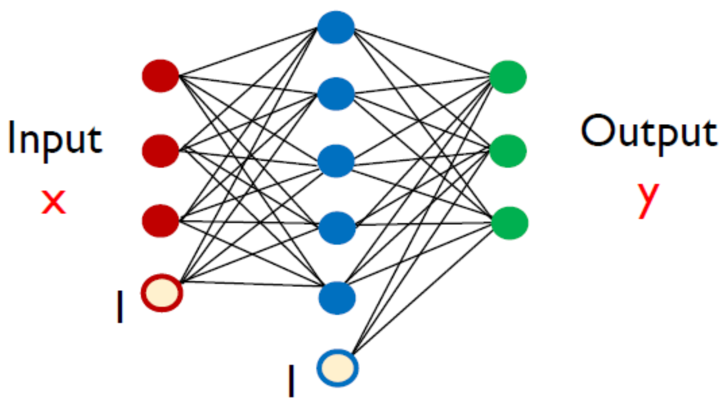
C_1 = **Handwritten 2**

C_2 = **Handwritten 4**

C_3 = **Handwritten 6**

C_4 = **Handwritten 8**





◆ Data

Input data are the extracted information from the picture of the hand-written numbers (**MNIST** Database). There are two data files offered: **Training _data_hw3.mat** file and one **Test_data1_hw3.mat** file.

Training _data_hw3.mat:

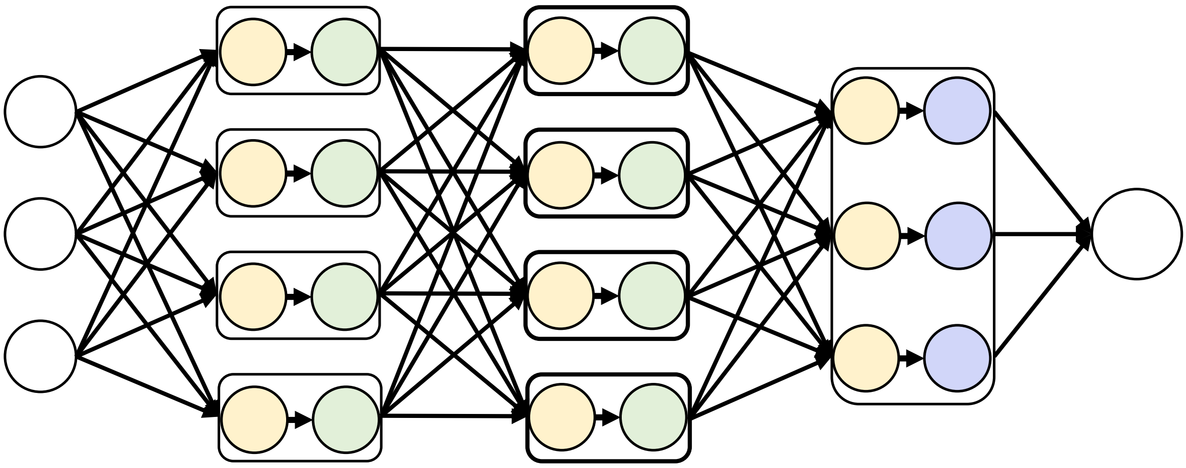
- **X_train** is a 4000x784 matrix. Every row corresponds to a 28x28 gray-scale image.
- **T_train** is a 4000x4 matrix, which records the target values of the training samples.

Test_data2_hw3.mat:

- **X_test** is a 2000x784 matrix. Every row corresponds to a 28x28 gray-scale image.

◆ Models

A simple neural network can be decomposed into the input layer, hidden layer(s) and output layer. A hidden layer is usually composed of an **inner-product layer** and an **activation layer**. Likewise, an output layer normally contains an **inner-product layer** and a **softmax layer**.



Let the m-by-1 vector $\boldsymbol{x}$ and n-by-1 vector $\boldsymbol{y}(\boldsymbol{x})$ be the input and output of a layer respectively, the forward-propagation and back-propagation can be inferred as bellow:

● Inner-product layer

✧ Forward-propagation

$$\boldsymbol{y}(\boldsymbol{x}) = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{w}_1^{\boldsymbol{T}}\boldsymbol{x} + b_1 \\ \vdots \\ \boldsymbol{w}_n^{\boldsymbol{T}}\boldsymbol{x} + b_n \end{bmatrix} = [\boldsymbol{w}_1 \dots \boldsymbol{w}_n] \cdot \boldsymbol{x} + \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix} = \boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{x} + \boldsymbol{b}$$

✧ **Back-propagation**

$$\nabla_x E = \begin{bmatrix} \frac{\partial E}{\partial x_1} \\ \vdots \\ \frac{\partial E}{\partial x_m} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial y_1}\frac{\partial y_1}{\partial x_1} + \cdots + \frac{\partial E}{\partial y_n}\frac{\partial y_n}{\partial x_1} \\ \vdots \\ \frac{\partial E}{\partial y_1}\frac{\partial y_1}{\partial x_m} + \cdots + \frac{\partial E}{\partial y_n}\frac{\partial y_n}{\partial x_m} \end{bmatrix} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_n}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_1}{\partial x_m} & \cdots & \frac{\partial y_n}{\partial x_m} \end{bmatrix} \begin{bmatrix} \frac{\partial E}{\partial y_1} \\ \vdots \\ \frac{\partial E}{\partial y_n} \end{bmatrix} = \boldsymbol{A} \begin{bmatrix} \frac{\partial E}{\partial y_1} \\ \vdots \\ \frac{\partial E}{\partial y_n} \end{bmatrix}$$

$$\nabla_{\boldsymbol{W}} E = \begin{bmatrix} \frac{\partial E}{\partial w_{11}} & \cdots & \frac{\partial E}{\partial w_{n1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial E}{\partial w_{1m}} & \cdots & \frac{\partial E}{\partial w_{nm}} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial y_1}\frac{\partial y_1}{\partial w_{11}} & \cdots & \frac{\partial E}{\partial y_n}\frac{\partial y_n}{\partial w_{n1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial E}{\partial y_1}\frac{\partial y_1}{\partial w_{1m}} & \cdots & \frac{\partial E}{\partial y_n}\frac{\partial y_n}{\partial w_{nm}} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial y_1} \\ \vdots \\ \frac{\partial E}{\partial y_n} \end{bmatrix} \boldsymbol{B}$$

$$\nabla_{\boldsymbol{b}} E = \begin{bmatrix} \frac{\partial E}{\partial b_1} \\ \vdots \\ \frac{\partial E}{\partial b_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial y_1}\frac{\partial y_1}{\partial b_1} \\ \vdots \\ \frac{\partial E}{\partial y_n}\frac{\partial y_n}{\partial b_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial y_1} \\ \vdots \\ \frac{\partial E}{\partial y_n} \end{bmatrix} \boldsymbol{C}$$

Please find the matrices **A**, ***B*** and **C** by yourself.

Activation layer – Sigmoid function

✧ **Forward-propagation**

$$\boldsymbol{y}(\boldsymbol{x}) = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1/(1+e^{-x_1}) \\ \vdots \\ 1/(1+e^{-x_n}) \end{bmatrix}$$

✧ **Back-propagation**

$$\nabla_x E = \begin{bmatrix} \frac{\partial E}{\partial x_1} \\ \vdots \\ \frac{\partial E}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial y_1}\frac{\partial y_1}{\partial x_1} \\ \vdots \\ \frac{\partial E}{\partial y_n}\frac{\partial y_n}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial y_1} \\ \vdots \\ \frac{\partial E}{\partial y_n} \end{bmatrix} \boldsymbol{D}$$

Please find the matrix ***D*** by yourself.

- **Activation layer – Rectified function**

  ✧ **Forward-propagation**

$$\boldsymbol{y}(\boldsymbol{x}) = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} [x_1 > 0] \cdot x_1 \\ \vdots \\ [x_n > 0] \cdot x_n \end{bmatrix}$$

  ✧ **Back-propagation**

$$\nabla_{\boldsymbol{x}} E = \begin{bmatrix} \frac{\partial E}{\partial x_1} \\ \vdots \\ \frac{\partial E}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial y_1}\frac{\partial y_1}{\partial x_1} \\ \vdots \\ \frac{\partial E}{\partial y_n}\frac{\partial y_n}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial y_1} \\ \vdots \\ \frac{\partial E}{\partial y_n} \end{bmatrix} \boldsymbol{E}$$

  Please find the matrix $\boldsymbol{E}$ by yourself.

- **Softmax layer**

  ✧ **Forward-propagation**

$$\boldsymbol{y}(\boldsymbol{x}) = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \frac{1}{\sum_{i=1}^{n} e^{x_i}} \begin{bmatrix} e^{x_1} \\ \vdots \\ e^{x_n} \end{bmatrix}$$

  ✧ **Back-propagation**

$$\nabla_{\boldsymbol{x}} E(\boldsymbol{y}, \boldsymbol{t}) = \begin{bmatrix} \frac{\partial E}{\partial x_1} \\ \vdots \\ \frac{\partial E}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial y_1}\frac{\partial y_1}{\partial x_1} \\ \vdots \\ \frac{\partial E}{\partial y_n}\frac{\partial y_n}{\partial x_n} \end{bmatrix} = \begin{bmatrix} y_1 - t_1 \\ \vdots \\ y_n - t_n \end{bmatrix} = \boldsymbol{y} - \boldsymbol{t}$$

There is only the inner-product layer containing parameters which need to be optimized. Please follow the **gradient descent** approach to iteratively update the parameters.

$$\boldsymbol{W}^{new} = \boldsymbol{W}^{old} - \eta \nabla_{\boldsymbol{W}} E$$

$$\boldsymbol{b}^{new} = \boldsymbol{b}^{old} - \eta \nabla_{\boldsymbol{b}} E$$

◆ Tasks

1. Layers as Matlab Functions

Implement the forward-propagation and back-propagation of every layer into Matlab functions **(LayerName)_ForProp(.)** and **(LayerName)_BackProp(.)**. You are suggested to modify the templates attached in the HW package to accomplish the task.

2. One-hidden-layer Neural Network

Based on the Matlab functions created in the task 1, you are able to build **one-hidden-layer NNs**. Please use such models to deal with the MNIST classification problem. (Choose the **sigmoid function** as the activation function.) Predict the 2000x4 target matrix **T_test** given the input data **X_test** and save it in the file **OneHidNN_T_test.mat**.

3. Two-hidden-layer Neural Network

One step further, you are asked to build **two-hidden-layer NNs**. Please use such models to deal with the MNIST classification problem. (Choose the **rectified function** as the activation function.) Predict the 2000x4 target matrix **T_test** given the input data **X_test** and save it in the file **TwoHidNN_T_test.mat**.

What should be uploaded?

- ☐ Your Matlab source code with comments.
- ☐ The **OneHidNN_T_test.mat** file which contains **T_test**.
- ☐ The **TwoHidNN_T_test.mat** file which contains **T_test**.
- ☐ The **ReadMe.txt** file which describes how to run your program.
- ☐ Your **report** in the format of .pdf or .doc.

Reminders:

- ☐ There won't be a need for demonstration.
- ☐ Please make sure your source code can be compiled by **Matlab**.
- ☐ **DO NOT COPY!!!（懶人包、考古題亦同）**