CSC173-CS4 Intelligent Systems
First Semester, SY 2021-2022

Assignment No. 2
Due: December 20, 2021 (Monday) at 11:55PM

I. About the Homework

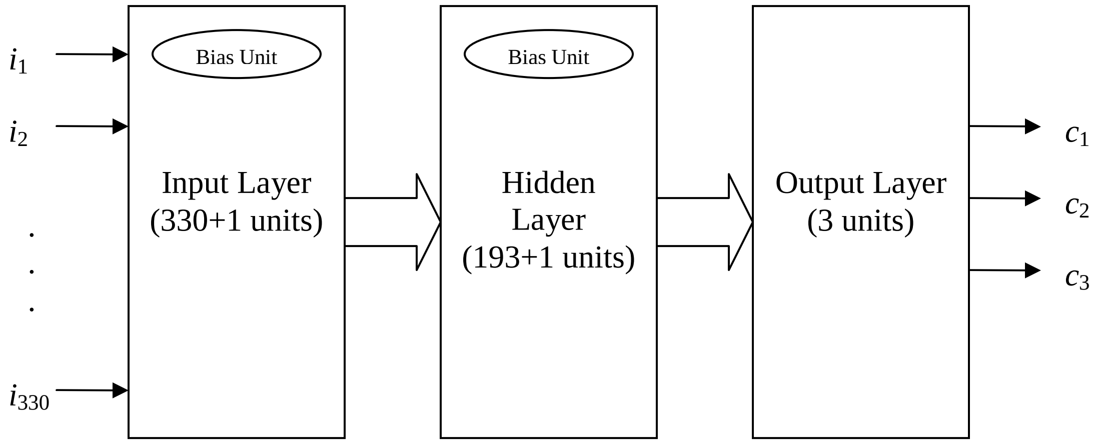
This homework is about artificial neural networks. You are going to implement an artificial neural network for classifying 15 by 22-pixel binary images representing the handwritten characters “c”, “e” and “d”. The parameters of this network, including the weights, will be specified in this homework.

II. Network Parameters

You are going to complete the Java source file named “`HW2RecognizerANN.java`” by implementing the `recognizerANN()` method. You may add your own methods in that source file. The `recognizerANN()` method takes one (1) parameter and returns an array of `double`s with three (3) elements. The input parameter is the input array of `double`s with $15 \times 22 = 330$ elements and this is where the input data for classification will be passed. Each element of the input array is either 0.0 or 1.0 depending on whether the corresponding pixel of the input image is white or black. If the input pixel is white, the corresponding element of the input array is 0.0. Otherwise, if the input pixel is black, the corresponding element of the input array is 1.0. The output array (the returned array) will have elements $[o_1, o_2, o_3]$ close to [1.0, 0.0, 0.0] if the input image is the letter “e”. The output array will be close to [0.0, 1.0, 0.0] if the input image is the letter “c” and will be close to [0.0, 0.0, 1.0] if the input image is the letter “d”. The input array $[i_1, i_2, i_3, ..., i_{330}]$ is mapped to the pixels of the input image as shown in the illustration below.

i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}	i_{15}
i_{16}	i_{17}	i_{18}	i_{19}	i_{20}	i_{21}	i_{22}	i_{23}	i_{24}	i_{25}	i_{26}	i_{27}	i_{28}	i_{29}	i_{30}
...														
i_{316}	i_{317}	i_{318}	i_{319}	i_{320}	i_{321}	i_{322}	i_{323}	i_{324}	i_{325}	i_{326}	i_{327}	i_{328}	i_{329}	i_{330}

The network is a three-layer feed-forward network with one (1) input layer, one (1) hidden layer, and one (1) output layer. Refer to the slides in class for the details on how artificial neural networks work. The input layer has 330 non-bias units that take the corresponding elements of the input array as input. The input weights in these units are 1.0 and the outputs are $a_j = i_j$, $1 \le j \le 330$ for the 330 units, that is, the activation function is $f(x) = x$. This layer also has a bias input unit (for the hidden layer) that takes no input and outputs $a_0 = 1.0$. The hidden layer has 193 non-bias units and 1 bias unit (for the output layer). The bias unit in this layer takes no input and outputs $b_0 = 1.0$. Each of the 193 non-bias units in the hidden layer takes 331 a_j’s, $0 \le j \le 330$ as inputs. The corresponding weights for these 331 inputs are $w_{j,k}$, $0 \le j \le 330$, $1 \le k \le 193$ for the connections from input *i* to unit *k*. The output of each unit *k* is $b_k = g(\sum_{j=0}^{330} a_j \times w_{j,k})$ where g is the activation function $g(x) = \frac{1}{1+e^{-x}}$. The output layer has 3 units and each unit in this layer takes the 194 outputs b_k, $0 \le k \le 193$, of the previous layer as inputs. The weights of the incoming connections to this layer are $v_{k,l}$, $0 \le k \le 193$, $1 \le l \le 3$ from input k to unit l. The output of each unit l is $c_l = g(\sum_{k=0}^{193} b_k \times v_{k,l})$ where g is the activation function defined previously. Shown below is a block diagram of the network.



The outputs c_l, $1 \le l \le 3$, of the network should be post-processed before returning them as the output of the `recognizerANN()` method. The elements o_l of the output array are computed as $o_l = t(s(c_l))$, where t and s are trimming and scaling functions defined below.

$$t(x) = \begin{cases} 0 & if\, x < 0 \\ 1 & if\, x > 1 \\ x & otherwise \end{cases} \qquad s(x) = \frac{x - 0.1}{0.8}$$

The weights $w_{j,k}$ and $v_{k,l}$ will be supplied in a text file named "`weights.txt`" included in this homework package.

III. The Weights File

The weights file contains two tables of weights. The first table has 193 rows and 331 columns. The lines of the file occupied by this table are lines 1 through 193. The second table has 3 rows and 194 columns. The lines of the file occupied by this table are lines 200 through 202. The tables contain real number entries in a format that is compatible with the Java programming language. The columns of the tables are aligned using space characters. Each row of the tables corresponds to the weights of the connections going into a single unit. That is, a row specifies the weights of the incoming connections into a unit starting from the first connection to the last connection in successive order. The first table specifies the weights of the incoming connections of the units in the hidden layer. The second table specifies the weights of the incoming connections of the units in the output layer. In terms of the weights $w_{j,k}$ and $v_{k,l}$ defined in the previous section, the entries of the tables are shown below.

First Table

$w_{0,1}$	$w_{1,1}$	$w_{2,1}$	…	$w_{330,1}$
$w_{0,2}$	$w_{1,2}$	$w_{2,2}$	…	$w_{330,2}$
		…		
$w_{0,193}$	$w_{1,193}$	$w_{2,193}$	…	$w_{330,193}$

Second Table

$v_{0,1}$	$v_{1,1}$	$v_{2,1}$	…	$v_{193,1}$
$v_{0,2}$	$v_{1,2}$	$v_{2,2}$	…	$v_{193,2}$
$v_{0,3}$	$v_{1,3}$	$v_{2,3}$	…	$v_{193,3}$

Due to the limitations of the Java programming language (which limits data per class to 64KB and bytecode per method to 64KB in each .class file), the tables cannot be hardcoded into a single class file. It is recommended that the weights be directly loaded from a file by the running program.

IV. Sample Output

To determine whether you implemented the network correctly or not, two sample images with their corresponding outputs are given below.

Image Pattern	o_1	o_2	o_3
Blank pattern (all white)	0.000000	1.000000	0.111851
Border pattern (blue around the borders, one (1) pixel thick)	0.000000	0.323043	1.000000

V. Compilation and Execution

To compile and/or run the sources in this homework, include the provided "`HW2.jar`" file as well as the current directory containing the sources in the classpath. This can be done by using the "`-cp [classpaths]`" option of the "`javac`" and "`java`" executables. For example, assuming that the "`HW2.jar`" file and the source files are in the current directory, use the following commands to compile the sources and run the program.

To compile the source files:

```
javac -cp HW2.jar;. Filename.java
```

Example:

```
javac -cp HW2.jar;. HW2RecognizerANN.java
```

To run:

```
java -cp HW2.jar;. HW2RecognizerANN
```

VI. Deliverables

You are required to submit to MOLE the whole homework package together with your modifications in a single .ZIP file.