CS 5000: Theory of Computability
Assignment 3

Vladimir Kulyukin
Department of Computer Science
Utah State University

September 16, 2017

Learning Objectives

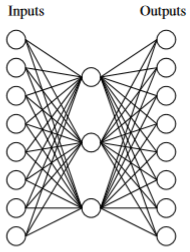
1. ANNs
2. Backpropagation

Problem 1 (5 points)

Implement the three-layer ANN shown in Fig. 1 from Chapter 4 of Tom Mitchell's text on machine learning. The ANN is trained to approximate the identity function that maps eight binary strings with exactly one 1 to themselves. Your implementation should include the class ANN.java or ANN.py. The class should implement three methods: build(), train(numIterations), and fit(input).

The build() method constructs the three-layer network in Fig. 1. The train(numIterations) method takes the number of iterations and uses the backpropagation algorithm to train the ANN for the specified number of iterations. You can also implement another version of this method, train(error), that keeps training the ANN until the error drops below the value specified by the parameter. Finally, the method fit(input) takes a 8-bit binary string as input and returns the output binary string, i.e., the values in the 8 output nodes. For simplicity, you can represent the inputs and outputs as integer arrays. But this is just a recommendation.

Fig. 2 shows a trained ANN for this problem from Chapter 4 of Mitchell's text. Your ANN's weight do not have to be exactly equal to these. So long

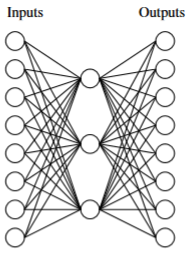


A target function:

Input		Output
10000000	$\rightarrow$	10000000
01000000	$\rightarrow$	01000000
00100000	$\rightarrow$	00100000
00010000	$\rightarrow$	00010000
00001000	$\rightarrow$	00001000
00000100	$\rightarrow$	00000100
00000010	$\rightarrow$	00000010
00000001	$\rightarrow$	00000001

Figure 1: ANN before training.

A network:



Learned hidden layer representation:

Input		Hidden Values				Output
10000000	→	.89	.04	.08	→	10000000
01000000	→	.01	.11	.88	→	01000000
00100000	→	.01	.97	.27	→	00100000
00010000	→	.99	.97	.71	→	00010000
00001000	→	.03	.05	.02	→	00001000
00000100	→	.22	.99	.99	→	00000100
00000010	→	.80	.01	.98	→	00000010
00000001	→	.60	.94	.01	→	00000001

Figure 2: ANN after training.

as your ANN maps each 8-binary string with exactly one 1 to itself, it is correct.

Also, implement the methods save() and restore(). The method save() should save the trained network into a file. If you use Python, you can use the pickle package. If you use JAVA, you may want to try ObjectOutputStream.writeObject() and ObjectInputStream.readObject().

What to Submit

Submit your ANN.py or ANN.java via Canvas.

Happy Hacking!