**Texas Tech University**

**Department of Computer Science**

**Course:** Introduction to Artificial Intelligence **Group:** 1

**Instructor:** Dr. Juan Carlos Rojas **Email:** [Juan-Carlos.Rojas@ttu.edu](mailto:Juan-Carlos.Rojas@ttu.edu)

**Hours:** 8:00 – 12:00 (Saturdays) **Room:** 320

# Homework 6

Due Saturday, July 6 at 8:00am.

## Practice 1

* Consider the 2-layer neural network for MNIST classification
  + From Mnist\_NN\_2.py
* Save a history of the four measurements at every epoch eval step:
  + Cost against training data
  + Cost against test data
  + Accuracy against training data
  + Accuracy against test data
* Hints:
  + Define a Python list for each, and append the values every time you compute them
* Make two plots using MatPlotLib
  + Cost vs. epoch (both training and test)
  + Accuracy vs. epoch (both training and test)

## 

## Optional Problem 1b

* Based on Mnist\_NN\_3.py
* Train a 2-layer neural network for MNIST with:
  + - 200 neurons
    - Learning rate = 0.05
    - 100 epochs
    - TanH activation function in the hidden layer
  + Graph the performance over time
  + Report the best test accuracy
* Change the activation function to ReLU
  + How do your results compare with the previous?
  + Did it train any faster?
* Now change to ELU
  + Did your results improve?
* There may be run-to-run variations
  + Run a few times to be sure of trends

## Practice 2

* Consider the 2-layer neural network for the classification of credit defaults
  + Based on CreditDefault\_NN\_2.py
* Change the hidden layer activation function to ReLU
* Test with different number of hidden neurons: 5, 10, 15, 20, 30, etc.
  + Did results improve, get worse, or are the same as with ELU?
  + Did it converge faster, slower or about the same?
* Suggest using:
  + Adam optimizer with initial learning rate = 0.001
  + 1000 epochs
* Note that there might be run-to-run variations

## Practice 3

* Pick one problem between MNIST or Vehicle Price
* Pick a number of hidden nodes that you think has a potential for improving through regularization
* Try both L1 and L2 regularization with different scale values
* Pick the best regularization value you can find
* Are your test results better than anything we have tried so far?

## Optional Problem 3b

* Repeat the regularization optimization of problem 3 but with different numbers of hidden nodes
* How do your results compare with the previous results without regularization?
* Is there an optimal number of nodes, or does it continue to be “the bigger the better”
* What is the best test score that you can find?