University of Houston
COSC 4371 Homework 3
Due: October 24, 2025 (11:59 PM)

R. Verma, B. Tuck and A. Dunbar

October 14, 2025

Instructions

- Submit a single PDF file containing all answers and required screenshots.
- Internet use is permitted for installing any libraries needed, and for downloading datasets from the UCI Machine Learning Repository.

Question I: Theoretical Concepts

1. **SVM Probability Output:** Can the SVM classifier we discussed in class be changed so that it outputs a probability distribution on the test dataset? That is, instead of just classifying the data, can it also output a probability value that gives an idea of the confidence the SVM model has in its classification? If yes, explain how it can be done. If not, justify why it cannot be done. Why would this be useful in cybersecurity?
2. **Classifier Comparison:** Compare naive Bayes, k-NN and decision tree classifiers with respect to complexity of training and testing, the assumptions made, the parameters involved, objective function, optimization technique, and expected dataset type(s).
3. **RNN vs CNN:** What are the main similarities and differences between the RNN and CNN models? Which one would you use for intrusion detection and why?

Question II: Neural Network Architecture

Task: Find the smallest neural network that can correctly classify the points shown in Figure 1.

Data Specification: Interval A is $[1/3, 2/3]$ and interval B is $[0, 1/3) \cup (2/3, 1]$. Create 80 random training data points (40 from class A and 40 from class B) and 20 test data points (10 from class A and 10 from class B). All points should be in the interval $[0, 1]$.

Requirements: Use fewer than 5 neurons in the hidden layer (there is no need for more than one hidden layer). Once you find the simplest architecture, comment on why a network that has one fewer neuron in the hidden layer cannot classify the picture correctly and show the maximum accuracy you get with both architectures.

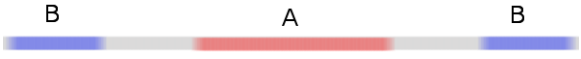

Figure 1: Classification problem for neural network design

Question III: Neural Network Implementation

Objective: Build a simple neural network from scratch using NumPy to understand gradient descent and backpropagation.

Implementation Tasks

Complete the TODO functions in `hw3_skeleton.py`, train with three learning rates (0.1, 0.5, 5.0), and generate learning curves and accuracy plots.

Deliverables

Submit working code (`hw3_skeleton.py`), two plots (learning curves and accuracy comparison), and final test accuracies for all three learning rates.

Analysis Questions

Answer in 2-3 sentences each:

1. Which learning rate performs best? Why?
2. What happens with the high learning rate? Explain the behavior.
3. What is the purpose of the validation set?
4. Explain how backpropagation computes gradients.

Question IV: Convolutional Neural Networks

1. **Convolution Operation:** Explain what a convolution is with respect to CNNs.
2. **CNN Applications:** Aside from text data, what kind of data are CNNs good at handling? Explain why they are effective in these domains.
3. **Text Data Preprocessing:** Text data has to be transformed before it works with a CNN. How is this accomplished?
4. **CNN vs Transformers:** CNNs, RNNs, and LSTMs ultimately fell out of favor for text data and are largely replaced by Transformers now. Aside from something simple like 'transformers work better,' why are they largely replaced by transformers?

Optional Exercises

Note: Same instructions as for previous homework about logging and submission.

1. **Semi-Supervised Learning:** Look for the fracpete/collective-classification-weka-package for Weka on GitHub. It extends Weka 3.7 with semi-supervised learning algorithms.

   Task: Try a semi-supervised learning algorithm by keeping aside 1000 randomly selected URLs from the dataset as unlabeled data (i.e., your algorithm should not make use of any label information for these 1000 URLs after selection). Select these 1000 URLs in the same proportion of the two classes as the entire dataset.

   Requirements: Explain how you ensured the correct proportion of the two classes in the "unlabeled dataset" and justify your choice of semi-supervised algorithm. Report average results and compare them with the results for the previous two questions. For every training fold, provide the same 1000 URLs set aside earlier as unlabeled data.

2. **SVM with Different Kernels:** On the (approximately) 11,000-instance phishing URL dataset from UCI, run an SVM classifier with 5-fold cross validation with at least two different kernels (no more than one can be a polynomial kernel).

Academic Honesty Policy

All sources must be properly cited in your answers and you must write the answers **on your own**, no matter who or what you consult.