Assignment 4

CMSC 678 — Introduction to Machine Learning

Due Tuesday April 23rd, 2019, 11:59 PM

Item	**Summary**
Assigned	Wednesday April 10th, 2019
Due	Tuesday April 23rd, 2019
Topic	Neural Networks
Points	105

In this assignment you will experiment with some small neural networks, become comfortable with the math of feed forward networks; and implement, experiment with, and compare neural network classifiers.

You are to *complete* this assignment on your own: that is, the code and writeup you submit must be entirely your own. However, you may discuss the assignment at a high level with other students or on the discussion board. Note at the top of your assignment who you discussed this with or what resources you used (beyond course staff, any course materials, or public Piazza discussions).

The following table gives the overall point breakdown for this assignment.

Question	1	2	3	4
Points	15	35	35	15

However, because this assignment handout *is* lengthy, I am first providing a **task list**. This task list captures the essence of the questions; it details, without other explanatory text, the tasks you are to do and what your completed assignment should answer. The task list enumerates what you must do, but it does not necessarily specify *how*—that's where the full questions come in.

Following the task list are the **full questions**. The full questions do *not* require you to answer additional questions, but they do provide specific details, hints, and explanations. You **should still read and reference** the full questions.

What To Turn In Turn in a **PDF** writeup that answers the questions; turn in all requested code **and models** necessary to replicate your results. Turn in models by serializing the learned models and parameters; your programming language likely natively supports this (e.g., Java's `Serializable` or Python's `pickle`). Be sure to include specific instructions on how to build/compile (if necessary) and run your code. Answer the following questions in long-form. Provide any necessary analyses and discussion of your results.

How To Submit Submit the assignment on the submission site:

https://www.csee.umbc.edu/courses/graduate/678/spring19/submit.

Be sure to select "`Assignment 4`."

Task List

1. (**20 points**) Explore the (in)famous XOR problem in Tensorflow Playground: `https://playground.tensorflow.org/#dataset=xor`.

   (a) Describe why it is called the XOR problem, and why it provides a problem to other classifiers we've studied. (Hint: look at the training data and consider how it relates to XOR.)

   (b) Notice how each of the neurons has some gradient shading. Describe what this shading represents. To help in this, feel free to change the various settings (activation, learning rate, features, etc.) and train the model (the big circular play button ▶).

   (c) Experiment with (at least) 16 configurations of a model; train each for at least 1,000 epochs and report on the final testing loss. Plot your results (in one or multiple readable plots) and discuss what you observed.

2. (**35 points**) Let each input x_i be a D-dimensional input vector, $x_i \in \mathbb{R}^D$ with corresponding K-dimensional one-hot label vector y_i^*. Your overall goal in this question is to derive the partial derivatives of the loss function for a feedforward multilayer perceptron to predict y_i from x_i with L hidden units. You may assume a cross-entropy loss function ℓ and that each hidden layer may have its own separate non-linearity (but that will be used consistently in computing that layer).

   (a) In [Eq-1], what function must $f^{(L+1)}$ be to use a cross-entropy loss function?

   (b) What are the parameters that must be learned?

   (c) If $D = 100, L = 2, K = 3$, and each *hidden* layer (non-input or output layer) has 50 neurons, how many parameters are there in total?

   (d) Let D, L, K, and the number of neurons (call it E) be unknown. For each variable $\clubsuit$, derive the partial derivate $\frac{\partial \ell}{\partial \clubsuit}$.

   (e) [Eq-1] does not include an explicit bias; describe how you can include the bias without changing [Eq-1] (or the derived gradients).

3. (**35 points**) Implement a multiclass neural network classifier with at least one hidden layer (you must implement the gradient yourself). Experiment and write a report:

   (a) discussing your implementation and convergence criteria.

   (b) discussing the concrete, quantifiable ways you validated your implementation was correct.

   (c) describing experiments with at least four different *configurations* and documenting the internal development progress through quantifiable and readable means, i.e., graphs and/or tables showing how different model configurations perform on `int-dev`. Provide your own analysis and summarization of this.

4. (**15 points**) Using the best neural configuration found in the previous question, and a *baseline of your choice*, train new models on the entire, original `train` set. At the end of this training, you should have two models trained on the entire 60,000 `train`ing set. Evaluate these models on the original 10,000 image `test` set. Include these evaluation results in your **report** and discuss the results.

This concludes the "task list." If this document only has 2 pages (and not 5 pages), please see the fully detailed PDF at `https://www.csee.umbc.edu/courses/graduate/678/spring19/materials/a4.pdf`. While the detailed PDF does not require any additional items to be completed, it specifies and clarifies many of the details.