Assignment #2 - Neural Networks

Context

This assignment is an opportunity to demonstrate your knowledge and practice solving problems about feedforward neural networks. This assignment contains an implementation component (i.e. you will be asked to write code to solve one or more problems) and an experiment component (i.e. you will be asked to experiment with your code and report results).

Logistics

Assignment due date: 2026-02-13

Assignments are to be submitted electronically through Brightspace. It is your responsibility to ensure that your assignment is submitted properly and that all the files for the assignment are included. Copying of assignments is NOT allowed. High-level discussion of assignment work with others is acceptable, but each individual or small group is expected to do the work themselves.

Implementation Part

Programming language: Python 3

For all parts of the implementation, you may use the Python Standard Library (https://docs.python.org/3/library/) and the following packages (and any packages they depend on): NumPy, Pandas, PyTorch, scikit-learn, SciPy. Unless explicitly indicated below or explicitly approved by the instructor, you may not use any additional packages.

You must implement your code yourself; do not copy-and-paste code from other sources. Please ensure your implementation follows the specifications provided; it may be tested on several different test cases for correctness. Please make sure your code is readable; it may also be manually assessed for correctness. You do not need to prove correctness of your implementation.

You must submit your implementation as a single file named "assignment2.py" with functions as described below (you may have other variables/functions/classes in your file). Attached is skeleton code indicating the format your implementation should take.

Experiments Part

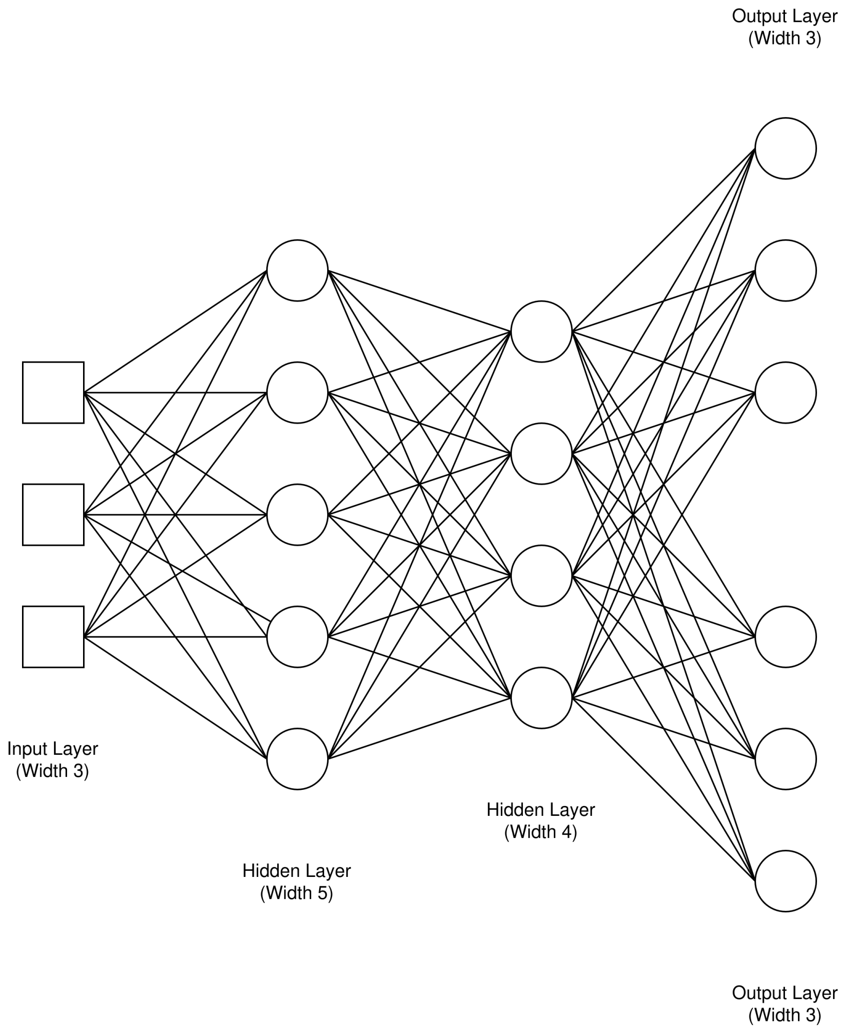
You must submit your report on experimental results as a single file named "assignment2.pdf".

Implementation Part

Question 1 [10 marks]

Consider the neural network illustrated in the figure below. It is designed for multi-task learning. Given an input vector, it produces predictions for two three-class classification tasks. Create a subclass of the "torch.nn.Module" class that implements a feedforward neural network with the architecture in the figure below. All hidden layers should use the ReLU activation function; the output layers should each use the softmax activation function.

The class must be named "MultitaskNetwork", and it must subclass "torch.nn.Module".



Question 2 [10 marks]

Complete the given code for the "multitask_training" function to train the network on training data. Assume the task and neural network architecture are the same as the above question. In particular, you must implement the following.

- A loss function, which is computed as a sum of categorical cross-entropy losses for each of the three-class classification tasks.
- An optimizer using stochastic gradient descent with a cosine learning rate schedule.

Once completed, your code for the "multitask_training" function should be runnable with the provided training data in the file "multitask_data.csv". The first three columns of the CSV file are a one-hot encoding of the first three-class classification task; the next three columns of the CSV file are a one-hot encoding of the second three-class classification task; the last three columns are the inputs. Each row in the file refers to an instance (i.e. an MLB position player).

The function should take one input argument: (1) the path to the file containing the training data.

The function should return one value: (1) a "MultitaskNetwork" object trained on the training data.

Question 3 [10 marks]

Consider developing a feedforward neural network for regression to predict the annual salary of a Major Leage Baseball (MLB) position player using information about their offensive, defensive, and baserunning statistics (e.g. hits, errors, stolen bases, etc.) and information about their contract status (e.g. free agency, arbitration, etc.).

A dataset for this task is provided in the attached "baseball.txt" file. Note that this dataset comes from one particular year, and it may not be sufficient for predicting salaries in different years. The first column of the file refers to the target we wish to predict (i.e. the salary of the MLB position player). The remaining columns of the file refer to features. Each row in the file refers to an instance (i.e. an MLB position player).

This dataset is publicly available here: https://www4.stat.ncsu.edu/~boos/var.select/baseball.html. Full details on the dataset are provided at this link.

Write a function that creates, trains, and evaluates a feedforward neural network to predict the salary of an MLB position player from the provided features. If necessary, you may implement pre-processing and/or post-processing of the data within this function.

The function must be called "mlb_position_player_salary".

The function should take one input argument: (1) the full file path to the "baseball.txt" dataset.

The function should return two values: (1) a PyTorch module object (subclassing "torch.nn.Module") that is trained to predict MLB position player salary period and (2) the performance of the model on the validation set.

Experiments Part

Question 4 [10 marks]

Conduct a series of experiments on the neural network you implemented in the above question to predict the salary of an MLB position player. For these experiments, hold out a test set.

a) Experiment on the number of neurons in each hidden layer. Plot the performance on the validation set as a function of the number of neurons in each hidden layer.
b) Experiment on the number of hidden layers in your network. Plot the performance on the validation set as a function of the number of hidden layers in your network.
c) Experiment on the number of epochs of training. Plot the performance on the validation set as a function of the number of epochs of training.
d) Experiment on the activation function. Plot the performance on the validation set for various common activation functions.

e) For the model with best performance on the validation set in the above experiments, report performance on the training set, validation set, and held out test set.