UNIVERSITY OF ARKANSAS

CSCE 5563: Introduction to Deep Learning

Homework Assignment #4

Submission Deadline: 11:59PM, 12/07/2022

What to submit:

- **1 pdf file** contains your solution and your name. Please don't submit the screenshot/scan of your handwritten solution. Exception is made for hand-drawn graphs. Grading rule: If it's unreadable, I won't grade. Confusing answer gets no point.
- **source code folder** that contains your code. The source code contains readme file on how to run the code.
- Put all the to-be-submitted files into a folder with the name format as:

  $\{LASTNAME\}\_\{FIRSTNAME\}$_homework1

What else to note:

A.. If you don't understand the question, ask the TA immediately at khoavoho@uark.edu or visit AICV Lab (JBHT #447) on Tuesday 1:45PM-2:45PM.

B.. You can discuss with your classmates, DO NOT COPY and please submit your own work.

C.. Solution for this homework is provided within 1-2 weeks after submission deadline or extended deadline, whichever comes later.

Question 1. (15 points): Generative Adversarial Networks (GANs)

(5 points): Although GANs have only been proposed in 2014, they have received a lot of attention and have been applied in many topics ranging from computer vision, natural language and signal processing. Please generally describe the required components in a GAN, how those components are trained, and how to deploy a GAN after we train it.

(5 points): A very common problem when training GANs is mode collapse. Please describe what this problem is, then, find and describe a solution to mitigate such problem.

(5 points): List three applications of GANs and name of the corresponding method of each application. For each application, please shortly describe its inputs, outputs, then briefly describe the method you chose to serve such application. Please also provide an URL to a scientific paper of the selected method.

Question 2. (20 points): Reinforcement Learning (RL)

(5 points): Value Iteration and Policy Iteration are basic algorithms to find an optimal policy π_* in Reinforcement Learning (RL). Please provide key points to distinguish between Value Iteration and Policy Iteration.

(5 points): Here is the Bellman equation defined during lecture:

$$V(s) = \max_a \sum_{s'} T(s,a,s')\Big[R(s,a,s') + \gamma V(s')\Big] \tag{1}$$

Prove the Bellman equation in Eq.1 is equivalent to Eq.2:

$$V(s) = \max_a \Big(R(s,a) + \gamma \sum_{s'} T(s,a,s')V(s')\Big) \tag{2}$$

(5 points): Please explain the role of γ in the above equation and indicate the valid range of γ's values. What happens if γ decreases its value. Vice versa, if γ grows its value, what would happen.

(5 points): What are Q function and V function in Reinforcement Learning (RL) ? When should we prefer Q function to V function and vice versa ? Please justify your answers.

Question 3. (65 points) Deep Q-Learning

Deep Q-Learning is a very popular method that can effectively train a neural network to solve RL tasks. Particularly, a neural network will be trained to implicitly map an input state observed from the RL environment to an action and corresponding Q-value. The network that is trained using deep Q-learning is so called Deep Q-Network (DQN).

In this question, you will have a chance to learn OpenAI's Gym library, a common open-sourced library that simulates many types of environments, from very simple ones like Cart Pole balancing to complicated Atari games, while making it super easy to interact with the environments.

Firstly, here is the URL to the colab notebook that you will be working on.

Your first two requirements are to complete the implementation of `class Network(nn.Module)`, this class is the DQN model that we intend to train:

- **TODO1 (10 points)**: Complete the declaration of our DQN's layers in `__init__ (in_dim, out_dim)`, which includes two hidden layers and an output layer:
  - Hidden Layer 1:
    - Input size: 'in_dim'
    - Output size: 128
    - Activation: ReLU
  - Hidden Layer 2:
    - Input size: 128
    - Output size: 128
    - Activation: ReLU
  - Output Layer:
    - Input size: 128
    - Output size: 'out_dim'
    - Activation: no activation function
- **TODO2 (15 points)**: Complete `forward` function by feeding input 'x' through the network defined in `__init__` and return the output of the network

Afterwards, you can run the whole program in the provided colab notebook. The program will train the DQN model in three separate environments, namely, CartPole, Acrobot, and LunarLander. Your next requirements are:

- **TODO3 (20 points)**: Referring to this URL, briefly describe each of the environments above. The description of an environment should show definitions of:
  - An input observation from the environment
  - An action to be fed back to the environment
  - Reward returned after executing a specific action

    - Conditions to terminate an episode

- **TODO4 (20 points)**: Report two plots for scores, losses (automatically updated during training) after you finish the execution of **every task** and submit the corresponding video rendered in the testing phase. Code for plotting is already included in the notebook, you do not have to implement.

Important Notice

1. You should use ipython notebook and Google Colab for **Question 3** to accelerate model training by GPU and have a clean, well-organized submission.

2. If you have any question, please feel free to contact TA (Khoa Vo) through email khoavoho@uark.edu, or visit AICV Lab (JBHT #447) on Tuesday 1:45PM-2:45PM.