FINAL PROJECT

Course: Introduction to Artificial Intelligence

Duration: 05 weeks

I. Formation

- The midterm project is conducted in groups of 04 – 05 students.
- Student groups conduct required tasks and submit the project following instructions.

II. Requirements

Students conduct the project using Google Colab and Python3 and submit a notebook when completed.

a) Task 1 (2.0 points): Constraint Satisfaction

YC1_1: Students implement the **EightQueenSolver** class to solve the problem of 8-Queen using the **backtracking** algorithm.

Class diagram of **EightQueenSolver**:

EightQueenSolver
+ EightQueenSolver() + solve(): void

in which, the solve() function runs **backtracking** algorithm and print down the result as a 2D-matrix on the screen with "Q" as a queen and "." as a blank cell.

Students arbitrarily add supporting attributes and methods.

Students implement a code block to demonstrate the operation of classes.

YC1_2: Students implement the **NQueenSolver** class to solve the problem of N-Queen using **backtracking** algorithm in general with N as the chess board size (N rows, N columns).

Class diagram of **NQueenSolver**:

NQueenSolver

- N: int
+ NQueenSolver(N: int) + solve(): void

in which, the solve() function run **backtracking** algorithm and print down the result as a 2D-matrix on the screen with "Q" as a queen and "." as a blank cell. If there does not exist a solution, then print down "UNSOLVABLE".

Students arbitrarily add supporting attributes and methods.

Students implement a code block to demonstrate the operation of classes.

b) Task 2 (2.0 points): Adversarial Search

Students implement the **MinimaxDecision** class as below.

MinimaxDecision
- root: Node - terminalStates: dict - successors: dict
+ MinimaxDecision() + read(filename: str): void + run(): void + print(): void

Node
- identifier: str - value: int
+ Node() + Node(identifier: str) + Node(identifier: str, value: int) + __str__(): str

YC2_1: Implement the two classes in which

- **terminalStates** is a dictionary,
  - key is a Node identifier, i.e., "n0", "n1".
  - value is an integer, i.e., 10, 20, 30.
- **successors** is a dictionary
  - key is a Node object
  - value is a list of Node objects,
  - For instance, n0:[n3, n5, n7] means that n3, n5, n7 are successors of n0.
- The **read**() function takes in a filename which is the path to a text file consisting of the structure of the adversarial search tree as below.

- The 1st line consists of two integers E and L. E is the number of branches and L is the number of terminal states/nodes.
- The next E lines consists of a string like "a b" per each, which means b is a successor of a.

  For example, **n1 n2** means n2 is a successor of n1.
- The next L lines consist of a string and an integer like "n v" per each, which means v is the utility of n.

  For example, **n20 30** means node n20 has the utility of 30.
- "n00" is root node.

YC2_2: Implement the **print**() function to print down all nodes in the adversarial search tree. For each node, display two attributes like (identifier, value). Students have to apply **recursion** for this function.

Hint: implement based on the **backtracking** algorithm.

YC2_3: Implement the **run**() function to find out and store values of all nodes in the tree using the **Minimax Decision** algorithm.

Students arbitrarily add supporting attributes and methods.

Students implement a code block to demonstrate the operation of classes.

c) Task 3 (2.0 points): Logical Agents

Students study and apply the **Glucose3** module to solve the problem of 8-Queen using propositional logic. Install the PySAT library as below.

```
pip install python-sat==0.1.7.dev12
```

Documentation of **Glucose3**: https://pysathq.github.io/docs/html/api/solvers.html

YC3_1: Students implement the **EightQueenSolver** class to solve the problem of 8-Queen using the **Glucose3** class and CNF clauses.

Class diagram of **EightQueenSolver**:

EightQueenSolver
+ EightQueenSolver() + solve(): void

in which, the solve() function runs the algorithm and print down the result as a 2D-matrix on the screen "Q" as a queen and "." as a blank cell.

Students arbitrarily add supporting attributes and methods.

Students implement a code block to demonstrate the operation of classes.

YC3_2: Students implement the **NQueenSolver** class to solve the problem of N-Queen in general using the **Glucose3** class and CNF clauses. N is the chess board size (N rows, N columns).

Class diagram of **NQueenSolver**:

NQueenSolver
- N: int
+ NQueenSolver(N: int) + solve(): void

in which, the solve() function runs the algorithm and print down the result as a 2D-matrix on the screen with "Q" as a queen and "." as a blank cell. If the does not exist a solution, then print down "UNSOLVABLE".

Students arbitrarily add supporting attributes and methods.

Students implement a code block to demonstrate the operation of classes.

Hint: 4 x 4 chess board (N = 4)

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

- Assign a positive integer to every cell (the integers are corresponding to propositional symbols)
- Traverse every cell to form sentences indicating constraints of the problem, i.e.,

cell 1 has a queen iff cells 2, 3, 4 do not have queens	$1 \Leftrightarrow -2 \wedge -3 \wedge -4$	(s1)
cell 1 has a queen iff cells 6, 11, 16 do not have queens	$1 \Leftrightarrow -6 \wedge -11 \wedge -16$	(s2)

cell 1 has a queen iff cells 5, 9, 13 do not have queens	1 ⇔ -5 ∧ -9 ∧ -13	(s3)

- Convert all sentences s1, s2, s3 to CNF clauses.
- After traversing all cells, find a model satisfying all CNF clauses using the Glucose3 class.

d) Task 4 (2.0 points): Machine Learning

Students study and apply the three libraries

- Decision Tree: https://scikit-learn.org/stable/modules/tree.html
- Naïve Bayes: https://scikit-learn.org/stable/modules/naive_bayes.html
- k-NN: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

to build up three models classifying hand-written characters in the MNIST dataset.

Students use the below code block to down the training and test sets.

```
from keras.datasets import mnist
(train_X, train_y), (test_X, test_y) = mnist.load_data()
```

YC4_1: Implement a Decision Tree model

- Fit data samples in the training set
- Compute the accuracies in the training and test sets
- Save the pre-trained model to file
- Load the model from a file
- Run inference (prediction) for at least 05 input samples
- Visualize the tree structure using tree.plot_tree()

YC4_2: Implement a Naïve Bayes classifier

- Fit data samples in the training set
- Compute the accuracies in the training and test sets
- Save the pre-trained model to file
- Load the model from a file
- Run inference (prediction) for at least 05 input samples

YC4_3: Implement a k-NN model

- Fit data samples in the training set
- Compute the accuracies in the training and test sets
- Save the pre-trained model to file
- Load the model from a file
- Run inference (prediction) for at least 05 input samples

YC4_4: Use the matplotlib.pyplot library to draw a twin-column bar chart to illustrate the accuracies in the training and test sets of the three designated models.

e) Task 5 (2.0 points): Report

- Student groups compose a presentation to report your work.
- **THERE IS NO PRESENTATION TEMPLATES. STUDENTS ARANGE CONTENTS IN A LOGICAL LAYOUT BY YOURSELVES.**
- The presentation must include below contents
    - Student list: Student ID, Full name, Email, Assigned tasks, Complete percentage.
    - Briefly present approaches to solve tasks, should make use of pseudo code/diagrams.
    - AVOID EMBEDDING RAW SOURCE CODE IN THE PRESENTATION.
    - Study topics are introduced briefly with practical examples.
    - Advantages versus disadvantages
    - A table of complete percentages for each task.
    - References are presented in IEEE format.
- **Format requirements:** slide ratio of 4x3, avoid using dark background/colorful shapes because of projector quality, students ensure contents are clear enough when printing the presentation in grayscale.
- Presentation duration is **05 minutes**.
- ***Instructions for the presentation recording are in the appendix.***

III. Submission Instructions

- Create a folder whose name is as

<Student ID 1>_< Student ID 2>_< Student ID 3>_< Student ID 4>

- Content:
  - **source** → source code folder (containing .ipynb files)
  - **presentation.pdf** → presentation.
  - **video.txt** → URL to the presentation recording.
- Compress the folder to a zip file and submit by the deadline.

IV. Policy

- **Student groups submitting late get 0.0 points for each member.**
- **Wrong student IDs in the submission filename cause 0.0 points for the corresponding students.**
- **Missing required materials in the submission loses at least 50% points of the presentation.**
- **Copying source code on the internet/other students, sharing your work with other groups, etc. cause 0.0 points for all related groups.**
- **If there exist any signs of illegal copying or sharing of the assignment, then extra interviews are conducted to verify student groups' work.**

-- THE END --

HƯỚNG DẪN
VIDEO THUYẾT TRÌNH
INSTRUCTIONS
FOR RECORDING PRESENTATION VIDEO

I. Mục tiêu/*Objectives*

- Nhóm sinh viên thực hiện quay video thuyết trình để báo cáo nội dung đồ án/đề tài.
  Student groups record a video to present your project/topic.
- Hình thức, công cụ, thời lượng được mô tả chi tiết ở mục tiếp theo.
  Formation, tools, and duration are described in the next section.

II. Yêu cầu/*Requirements*

- Công cụ: Zoom/Google Meet
  Tools: Zoom/Google Meet
- Thời lượng: tối đa 05 phút
  Duration: 05 minutes (no longer than 05 minutes)
- Hình thức:
  Formation
  - Nhóm sinh viên tạo một meeting để tham gia thuyết trình.
    Student groups create a meeting to present your work.
  - Đặt tên hiển thị theo dạng <MSSV>_<Họ tên>,
    Set the display name as <Student ID>_<Full Name>
    ví dụ 52200001_Nguyễn Văn A
    for example, 52200001_Nguyen Van A
  - Tất cả thành viên phải bật camera trong toàn bộ buổi thuyết trình.
    Every member must turn on your camera during the presentation.
  - Sinh viên trình chiếu bài thuyết trình nhưng phải đảm bảo hiển thị đầy đủ khuôn mặt của các thành viên còn lại.

Students show your presentation but ensuring to display all member facial thumbnails.

- Các sinh viên thay phiên trình bày các nội dung.
  Students share the role of presenting.
- Bài thuyết trình được quay lại, đảm bảo chất lượng hình ảnh và âm thanh.
  The presentation is recorded completely.

III. Hướng dẫn nộp bài/Submission Instructions

- Video tải lên Youtube và đặt ở chế độ "unlisted", tuyệt đối không để dạng "public".
  The recording is uploaded to Youtube with the "unlisted" sharing option. Do not share with the "public" option.
- Đặt tên theo cú pháp
  Rename the recording as below

  <Năm học>-<Học kỳ>-<Môn học>-<Tên nhóm>

  <Year>-<Semester>-<Course>-<Group Name>

  trong đó gồm:
  - <Năm học> theo dạng YYYY, ví dụ 2223, 2324, 2425
    <Year> in form of YYYY, for example, 2223, 2324, 2425
  - <Học kỳ> là "HK1" hoặc "HK2"
    <Semester> is "HK1" (Term 1) or "HK2" (Term 2)
  - <Môn học> là "AI" (NM Trí tuệ Nhân tạo) hoặc "MMDS" (Xử lý Dữ liệu lớn)
    <Course> is "AI" (Introduction to AI) or "MMDS" (Mining Massive Datasets)
  - <Tên nhóm> theo tên đã đăng ký
    <Group Name> as in registration
- Một thành viên nhóm đại diện nộp đường dẫn theo deadline được cho.
  Only one representative student submits the video URL by the deadline.

-- HẾT --

-- END --