ADDIS ABABA INSTITUTE OF TECHNOLOGY

ARTIFICIAL INTELLEGENCE ASSIGNMENT

Name: SOSNA WORKU
ID: UGR/3804/12

1) Identify two very different existing AI systems and characterize them based on the PEAS problem formulation. Give a detailed explanation of the applications based on these four fundamental concepts.

The two existing systems that I choose are smart assistance and diseases mapping.

Smart Assistance

Smart assistance - are devices loaded with software that are used to access information, perform tasks and control other devices. We can have a smart assistant on your computer or your phone, but most people use them through a smart speaker. The most known smart assistances include Alexa, Siri, Cortana and Google Assistant.

- ❖ Characterization of smart assistance based on PEAS

1) Performance measure: one of the performance measure for smart assistances or virtual assistances is giving instant answer for the request made by the user. Whenever the user request for assistance via the sensors fast and timely response must be provided. The other one is automated booking of orders for example, Google Duplex, Google's new human-sounding artificial intelligence. Duplex will soon let you book restaurant reservations and hair appointments. Duplex will call the restaurant for you and speak with a human on the other end. The third one, lowered customer service costs. Lowered customer service costs is quality of service your customer received, how long it takes for you to answer their phone call, or just their overall experience with your brand. Automation makes customer service better at times and hence, we recommend a pick and choose method for businesses. But the cost must be minimized as much as possible and this is achieved considerably by smart or virtual assistances. The last one increased customer engagement and elimination of human errors.

2) Environment: Smart Assistants are tools that are installed on a PC or a mobile which means their working environment is a software. They can help to diagnose and resolve issues, manage data, reinstall device software, and much more. Smart assistant software agents are used to perform task or service for individual. Sometimes the term ‘Chabot’s used to refer smart assistant, which is a new technology that would harnessed to create an intelligent virtual personal assistant which focus on user based information. The software focuses on virtual assistant and structural elements of a virtual assistant system.

3) <u>Actuator:</u> The actuators available to smart assistances are Speakers, Microphones, and Display items. It will output to a display screen or voice synthesizer talk back to customers/peoples and perhaps some way to communicate with other devices.

4) <u>Sensors:</u> The sensors available to smart assistances are Camera, Speaker recognition (voice recognition), Text input. For instance Alexa uses the fourth generation Echo smart speaker, which has a temperature sensor built in, as do Echo Show displays and Echo Plus speakers. These can be used to trigger Alexa routines when the temperature in a room reaches a set point.

<u>Disease Mapping</u>

Disease mapping – is the use of artificial intelligence (AI) to scan social media, news reports, internet search queries, and other information streams for signs of disease outbreaks. The international alarm about the COVID-19 pandemic was sounded first not by a human, but by a computer.

- ❖ Characterization of disease mapping based on PEAS

1) <u>Performance measure:</u> AI diseases mapping makes data mapping fast and accurate as data accuracy is one of the performance measure for disease mapping. It employs machine learning for inferring data mapping predictions from existing library of tested and validated data maps that dramatically reduces the effort and time required to create intelligent data mappings. Some of the disease that accurately detected by the AI systems are finding indicators of diabetic retinopathy in eye images, and Assessing the risk of sudden cardiac death or other heart diseases based on electrocardiograms and cardiac MRI images.

2) <u>Environment</u>: The working environments for disease mapping are Health data centers, Disease control and prevention, Research centers, and Maps. One of the advantages of disease mapping is disease prediction. Disease prediction using Machine Learning is the system that is used to predict the diseases from the symptoms which are given by the patients or any user. The system processes the symptoms provided by the user as input and gives the output as the probability of the disease. Human studies have shown how social media data could be used to map infectious diseases that have not yet been identified by public health institutions. Using unsupervised machine learning model.

3) <u>Actuator:</u> The actuator for disease mapping are the digital displays that are used to display the map.

4) Sensor: The sensor for disease mapping are the cameras, GPS that are used to take the data and act accordingly. The international alarm about the COVID-19 pandemic was sounded first not by a human, but by a computer. HealthMap, a website run by Boston Children's Hospital, uses artificial intelligence to scan social media, news reports, internet search queries, and other information streams for signs of disease outbreaks. Researchers have applied AI to automatically recognizing complex patterns in imaging data and providing quantitative assessments of radiographic characteristics.

Agent	Performance Measure	Environment	Actuator	Sensor
Smart Assistant	Instant answer, Automated bookings of orders, Lowered customer service costs, Increased customer engagement, Elimination of human errors	software	Speakers, Microphones, Digital display items	Camera, Speaker recognition (voice recognition), Text input
Disease Mapping	Data accuracy, Showing disease prevalence rate, Easy availability of the data needed,	Health data centers, Disease control and prevention, Research centers, Maps	Digital displays	Camera, GPS

Table: Conclusion for characterization of smart assistance and disease mapping (PEAS)

PEAS description of task environment

Task environment	Observable	Agent	Deterministic	Episodic	Static	Discrete
Smart assistance	Partially observable	Multi-agent	Stochastic	Sequential	Dynamic	Discrete
Disease mapping	Observable	Multi-agent	Stochastic	Episodic	Dynamic	Discrete

2) The text file that contains the graph is file.txt and the python file question2.py

3) I have wrote the functions breadth first search, depth first search, Dijkastra's algorithm and A* algorithm. They are written on question3.py file.

   The following table shows the benchmark in evaluating each of the algorithms. The average time required and solution length required for searching via each algorithm.

Algorithm	Average time required (in seconds)	Solution length
BFS	10.06×10^{-4}	20
DFS	9.96×10^{-4}	20
Dijkastra	1.69×10^{-2}	6
A*	1.79×10^{-2}	6

Table2: benchmark result for each algorithm

Discussion

BFS uses queue to get the adjacent nodes, but DFS uses stack which in turn results traversing of the adjacent node according to tree levels and tree depth. I almost get the same result in BFS and DFS as shown in the above table. A DFS algorithm may have a quick result if the goal is in the first branch than A* and DFS.

Dijkastra algorithm is the same as BFS with priory queue and is faster than BFS if we consider the weights of all but if the weights are not considered BFS is faster than Dijkastra's algorithm.

As we can see from the above table both Dijkastra and A* have the nearly the same result. Dijkastra and A* are almost the same, but A* finds the nodes using heuristic function which means it is an informed search. Which gives precedence for the nodes which have better cost than other nodes.

Sample results from the execution of the algorithms that and I took the averages for each of the algorithms to fill the above table (benchmark result for each algorithm).

```
============= RESTART: C:\Users\HP\Desktop\UGR_3804_12\question3.py ============
breadth first search
eforie -> oradea -> zerind -> sibiu -> arad -> timisoara -> lugoj -> mehadia ->
drobeta -> craiova -> rimnicu_vilcea -> pitesti -> faragas -> bucharest -> giurg
iu -> urziceni -> hirsova -> vaslui -> iasi -> neamt


$$$$$$ time $$$$$$
required time for breadth first search = 0.0 X 10^-4 second


###############
depth first search
eforie -> hirsova -> urziceni -> vaslui -> iasi -> neamt -> bucharest -> giurgiu
 -> faragas -> sibiu -> rimnicu_vilcea -> craiova -> drobeta -> mehadia -> lugoj
 -> timisoara -> arad -> zerind -> oradea -> pitesti


$$$$$$ time $$$$$$
required time for depth first search = 0.0 X 10^-4 second


###############
djkastra
eforie<->hirsova<->urziceni<->vaslui<->iasi<->neamt
The time requred for dijkastra algorithm 0.004307599971070886 seconds


###############
A* shortest path search algorithm
eforie<->hirsova<->urziceni<->vaslui<->iasi<->neamt
The time requred for A* algorithm 0.005716699990443885 seconds
```

```
============= RESTART: C:\Users\HP\Desktop\UGR_3804_12\question3.py ============
breadth first search
eforie -> oradea -> zerind -> sibiu -> arad -> timisoara -> lugoj -> mehadia ->
drobeta -> craiova -> rimnicu_vilcea -> pitesti -> faragas -> bucharest -> giurg
iu -> urziceni -> hirsova -> vaslui -> iasi -> neamt


$$$$$$ time $$$$$$
required time for breadth first search = 0.0 X 10^-4 second


###############
depth first search
eforie -> hirsova -> urziceni -> vaslui -> iasi -> neamt -> bucharest -> giurgiu
 -> faragas -> sibiu -> rimnicu_vilcea -> craiova -> drobeta -> mehadia -> lugoj
 -> timisoara -> arad -> zerind -> oradea -> pitesti


$$$$$$ time $$$$$$
required time for depth first search = 0.0 X 10^-4 second


###############
djkastra
eforie<->hirsova<->urziceni<->vaslui<->iasi<->neamt
The time requred for dijkastra algorithm 0.0070711999433115125 seconds


###############
A* shortest path search algorithm
eforie<->hirsova<->urziceni<->vaslui<->iasi<->neamt
The time requred for A* algorithm 0.007696899934671819 seconds
>>
```

```
breadth first search
eforie -> oradea -> zerind -> sibiu -> arad -> timisoara -> lugoj -> mehadia ->
drobeta -> craiova -> rimnicu_vilcea -> pitesti -> faragas -> bucharest -> giurg
iu -> urziceni -> hirsova -> vaslui -> iasi -> neamt


$$$$$$ time $$$$$$
required time for breadth first search = 0.0 X 10^-4 second


###############
depth first search
eforie -> hirsova -> urziceni -> vaslui -> iasi -> neamt -> bucharest -> giurgiu
 -> faragas -> sibiu -> rimnicu_vilcea -> craiova -> drobeta -> mehadia -> lugoj
 -> timisoara -> arad -> zerind -> oradea -> pitesti


$$$$$$ time $$$$$$
required time for depth first search = 0.0 X 10^-4 second


###############
djkastra
eforie<->hirsova<->urziceni<->vaslui<->iasi<->neamt
The time requred for dijkastra algorithm 0.007537900045359135 seconds


###############
A* shortest path search algorithm
eforie<->hirsova<->urziceni<->vaslui<->iasi<->neamt
The time requred for A* algorithm 0.009813599986955523 seconds
```