**Report**

**Project 4: The Imitation Game**

Parvathi Mahesh Hedathri: pm850

Kavya Kavuri: kk1069

Harish Udhayakumar: hu33

December 15 2021

1. How should the state space (current information) and action space (action selected) be represented for the model? How does it capture the relevant information, in a relevant way, for your model space? One thing to consider here is local vs global information.

* The state space can be represented in two forms as follows:

**Local information:**

We capture the information of agent and its immediate neighbors and decide on which node to move next.

Advantage:

* The data for training of the Neural Network doesn’t need to have all the combinations of agent position, blocked and free nodes. This consecutively reduces massive dataset size to train the Neural Network.
* This 3 x 3 local grid can be utilized to facilitate the agent to traverse through a grid of any N x N matrix.

Disadvantage:

* The information is local and hence, this the agent can only decide its next action based on its neighbors regardless of target’s position.

In the example shown below (a sample representation),

![Shape, square

Description automatically generated]() Represents current position of Agent in the local grid

![Shape, square

Description automatically generated]() Represents visited cells

![Shape

Description automatically generated with medium confidence]() Represents blocked cells

Chart

Description automatically generated

At every iteration we check the grid window of 3 x 3 with agent in the center. However, if the agent is present in the edges of grid world, then the agent is also represented to be in the edge of the local 3 x 3 grid.

How does this impact the agent?

If at a particular situation network predicts and decides to move downwards to next cell and in the next iteration, if the network predicts to move upwards, the agent might have to move up and down in an infinite loop.

Also, since the agent doesn’t have any information on the location of target at any point in time, the actions taken might not be necessarily in the direction of the target.

* Global Information:

We capture the information of entire grid and decide how should the agent move towards target.

Advantages:

* The Neural Network precisely predicts the next action of the agent towards the target, for a solvable maze.

Disadvantages:

* For precise prediction, the neural network must be trained with all possible combinations of agent’s current position, agent’s observed grid world (blocked and free nodes) and the target.
* For higher dimensions of grid world, the neural network demands a huge dataset.
* A dataset for N x N grid could only be used to solve N dimension test grid unlike the local grid which can be used for grid of any dimension.

The state space is a grid with information that is observed by agent. The information is represented as follows:

For local information: It is a 3 x 3 grid and for a global information it is an entire grid used to represent the state information at that instance.

The grid has following values in each cell that represents one particular state of that cell.

1 – Free and visited

2 – Blocked and visited

3 – Unvisited

4 – Current position of agent

Example of a local grid would look like:

In this scenario at this iteration, the information that the agent has is as follows:

* The agent is at the last row (nth row)
* The agent has 5 neighbors, and it is in the bottom most row of the grid world.
* The neighbor above is visited and free
* The neighbor to the left is blocked.
* The Neural Network now ideally predicts to move right.

The action space is a one-hot encoding vector of (1 x 4) dimension that represents the following:

[1, 0, 0, 0] = UP

[0, 1, 0, 0] = DOWN

[0, 0, 1, 0] = LEFT

[0, 0, 0, 1] = RIGHT

1. How are you defining your loss function when training your model?

* We use crossentropy loss function when there are two or more label classes. We expect labels to be provided in a one\_hot representation. Since we want to provide labels as integers, we use SparseCategoricalCrossentropy loss.

1. In training, how many episodes on how many different grid worlds were necessary to get good performance of your model on the training data?

* The neural network required an overall 127 episodes of 456 different grid worlds to get a 95% accuracy of the model on training data.

1. How did you avoid overfitting? Since you want the ML agent to mimic the original agent, should you avoid

overfitting?

* The **number of epochs** had to be optimal to avoid over fitting. Initially, as the number of epochs increased the accuracy in validating the model increased. However, this led to overfitting of model and the network failed to navigate the agent to reach target. The reduction in number of epochs, facilitated in avoiding overfitting the model.

We introduced **early stopping**  to avoid overfitting

Added dropout layer with 0.5 probability.

L2 regularization

1. How did you explore the architecture space, and test the different possibilities to and the best architecture?

Added different layers to the model with different activation function, changed number of nodes, tried different loss functions, trained over multiple epochs and batch size

1. Do you think increasing the size or complexity of your model would offer any improvements? Why or why not?

No, if we increase the size or complexity of the model then we tend to overfit the model and increase the variance for testing data so our testing data will have lower accuracy.

1. Does good performance on test data correlate with good performance in practice? Simulate the performance of your ML agent on new gridworlds to evaluate this.
2. For your best model structure, for each architecture, plot a) performance on test data as a function of training rounds, and b) average performance in practice on new gridworlds. How do your ML agents stack up against the original agents? Do either ML agents over an advantage in terms of training time?

Sample data of average trajectories of Mazes

|  |  |
| --- | --- |
| **Trajectory\_path of agent 1 (50 x 50)** | **Trajectory\_path of ANN (50 x 50)** |
| 244.4751 | 280.783739 |
| 326.929918 | 341.584636 |
| 209.441597 | 232.285295 |
| 226.880319 | 296.893381 |
| 242.606407 | 325.725015 |
| 324.154622 | 328.732751 |
| 257.54941 | 377.306129 |
| 239.764945 | 284.120274 |

Chart, bar chart

Description automatically generated

Note: Graph shows few points of Accuracy VS Epoch ( For Visualization purpose)