**AI Group**

Project 1 Fashion MNIST classification

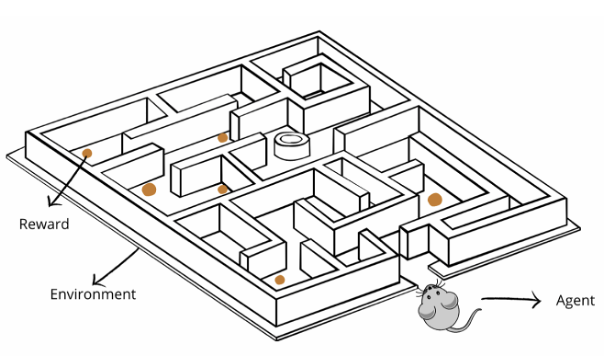
Build a convolutional neural network to do classification task on Fashion MNIST dataset, the dataset<https://github.com/zalandoresearch/fashion-mnist> , （1）try your best to improve the network’s classification performance, such as tune the learning rate, design network structures, augment the data et al; (2) visualize the network’s learned kernels to see what it has learned, visualize and try to understand the network’s hidden representations .

Project 2 Natural gradient descent method

Build a simple multilayer perceptron, use the natural gradient descent method to train the network and solve the IRIS flower classification problem, (dataset<https://www.kaggle.com/arshid/iris-flower-dataset>). (1) Train the network using natural gradient descent method, plot the learning curve for the IRIS problem and compare it with stochastic gradient descent method to understand Fisher-efficiency. (2) Computing the inverse of Fisher Information Matrix is difficult and many approximate methods have been proposed, pick some methods and have a try. (Reference paper: Adaptive natural gradient learning algorithms for various stochastic models, Fisher Information and Natural Gradient Learning in Random Deep Networks)

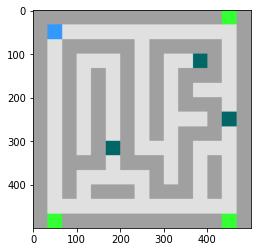
Project 3 Reinforcement Learning

Use Reinforcement Learning algorithms to simulate a rat in a maze. The rat learns to forage food effortlessly.



We will provide the environment class in a Jupyter Notebook. The environment class is responsible for responding to the agent’s action with reward/penalty, and will inform the agent if it has reached exit. You will implement the learning algorithm that tells the agent how to move to collect the most reward.

Concretely, in the following maze environment, an agent starts at the blue spot, and if it reaches any one of the green spots (exit/terminal), the trial is over. Three dark green spots represent three reward locations. Each time the agent reaches any one of these reward locations, it receives a large reward, and the reward is removed. To encourage the agent to finish the maze efficiently, it also receives a small penalty at every time step.



Suggestions: 1) Explore how to represent states such that they have Markov property; 2) Implement different RL learning algorithms in this task and compare their efficiency and differences; 3) Explore how Q values evolve during the training process; 4) Compare the epsilon-greedy policy with other stochastic policies.

**CN Group**

***1, Conductance-based thalamic reticular nucleus neuron model***

Build a conductance-based thalamic reticular nucleus neuron model according to paper [1].

* Step 1: implement the neuron model
* Step 2: reproduce the firing patterns of tonic firing and rebound bursting as seen in vivo TRN

[1] Li, Guoshi, Craig S. Henriquez, and Flavio Fröhlich. "Unified thalamic model generates multiple distinct oscillations with state-dependent entrainment by stimulation." PLoS computational biology 13.10 (2017): e1005797.

***2, Bifurcation analysis of FitzHugh-Nagumo model***

Make bifurcation analysis of the FitzHugh-Nagumo model. Try to understand the effect of various parameters in the model.

***3, Spiking circuit model for working memory***

Building up a spiking circuit model for working memory (Misha, Science, 2008).

* Step 1: build LIF neuron, voltage jump synapse, STP synapse
* Step 2: consider the connectivity used in the paper, build up the network model
* Step 3: try to encode one item in the model (Fig. 2)
* Step 4: try to encode two items in the network (Fig. 3)

***4, Spiking circuit model for decision making***

Building up a spiking circuit model for decision making (Xiaojing Wang, 2002), and comparing it with the rate model (Xiajing Wang, 2006).

* Step 1: build LIF neuron, NMDA synapse, AMPA synapse, GABAa synapse
* Step 2: consider the connectivity used in the paper, build up the network model
* Step 3: reproduce the decision dynamics with inputs of different coherence (Fig. 3 and Fig. 4 in *Xiaojing Wang, 2002*)
* Step 4: compare the reaction time of the spiking model and the rate model (Fig. 3 in *Xiaojing Wang, 2006*)

***5, CANN for anticipative tracking in 2D space***

Building a 2D CANN model for anticipative tracking.

* Step 1: build a 2D CANN model with periodic boundaries.
* Step 2: add spike frequency adaptation to introduce intrinsic dynamics
* Step 3: tuning the parameters to anticipative tracking a input formed by a Gaussian bump
* Step 4: anticipative tracking a real world object, e..g, a moving ball on a ground.