Adversary Task Simulation Model

Version 1.0 - 06/10/2018

A general overview of the program and how to use it can be found in the readme file of this repository.

Things that are different from the original thesis branch:

This simulation model is identical to the original thesis branch in all aspects except for the following: This model introduces adversary agents whose goal is to attack the swarm agents. The reward function of the leader's neural network has been modified to increase the error when agents are eaten, and reduce the error when agents reach the goal area without being eaten.

Current state of the simulation:

This simulation model currently compiles, runs, and improves over time when in the training mode. When running, the simulation outputs to the command line errors after each round, a rolling tally of how the model is performing after each epoch, and an average error for the entire runtime. These metrics are intended to give the user a better understanding of how the model is performing in real time.

Analysis of current model performance:

The model performs much better with higher percentages of agents being leaders, compared to lower percentages of the agents being leaders. The most tested and best performing percentage of leader agents was 40%, in which the simulation learned how to avoid the adversaries and herd the swarm of non-leader agents to the goal area safely nearly every round. 200 epochs were used to train these models with visualization disabled. For 50 agents, the training time on a 4th generation Intel i7 processor, clocked at 3.4Ghz, in a laptop, took around one hour to complete (single threaded). For 100 agents, the training time on a 4th generation Intel i7 processor, clocked at 3.4Ghz, in a laptop, took around two hours to complete (single threaded).

Some good performance was occasionally observed with 20% of the swarm being leaders, but the model did not perform reliably well. Poor performance was observed with percentages lower than 20% leaders.

Saved experiments:

Currently, there are saved weights from 6 different experiments in the /data/weights directory of this repository, for which the following experiments were conducted. Two batches of three swarms were trained. One batch, with 50 total agents, and the other batch with 100 total agents. Each batch was broken down into three ratios, each with an experiment of 10%, 20% and 40% of the agents being leaders. The following command line arguments are noted for better visualization (omitting "TRAIN" and executable file path):

5 50 200 1
10 50 200 1
20 50 200 1

10 100 200 1
20 100 200 1
40 100 200 1

To test any of these pre-trained models, simply ensure that the file path in the includes.h header file directs the program to the correct directory to load the weights upon testing (they are currently saved in the folder **/June_6_Experiments_in_report**). Use the “TEST” parameter to test the models. Using the “TRAIN” parameter will overwrite these experiments.

Example to test a pre-trained model from above: **./bin/main “TEST” 40 100 200 1**

Remaining issues and suggestions for improvement:
The current state of this simulation task is far from perfect, leaving room for future development and improvement to enable better performance of trained models while utilizing smaller leader ratios in the swarm. Below are issues with the simulation task which negatively impact the performance, learning ability, and analogy to real life scenarios:

- Adversaries only follow the first leader agents that spawns, ignoring all other agents.
- The number of epochs for which the models are trained has been picked ambiguously.
- The entire simulation only utilizes one core of the processor it is running on. GPU support or multi-threaded processor support would greatly reduce training time.
- The error function’s values have been picked somewhat randomly, as the current values gave the best results during the simulation’s development in the Spring term of 2018.
- Cases which the error function considers could also be added/omitted to increase the desire performance.
- It is recommended to add a reward for keeping non-leader agents at the goal.
- The adversary agents are too slow to be competitive with the other agents.
- Agents are allowed to overlap and be in the same place at the same time in some cases, which does not reflect a real scenario.