A Series of Unlikely Events: Learning Patterns by Observing Sequential Behavior

Introduction

Modeling patterns of behavior is a task that underlies numerous difficult artificial intelligence tasks:

How do I detect when adversaries are deviating from normal routines?

How can I automate the teaching of novice analysts to perform complex tasks as if they were expert analysts?

In this work, we use a class of techniques called **Inverse Reinforcement Learning (IRL)** to model sequential behavior to answer questions like these and others.

Methodology

Given observations of behavior:

$$\mathcal{B}=\{((s_1,a_1),(s_2,a_2),\ldots)_1,\ldots,((s_1,a_1)\ldots)_n\}$$

Learn a reward $R:\mathcal{S}\ \mathrm{x}\ \mathcal{A}\mapsto\mathbb{R}$ that best explains the observed behaviors.

How IRL algorithms work:

(Two steps)

1. Learn policy $\pi:\mathcal{S}\mapsto\mathcal{A}$ from reward R (policy trained to maximize expected reward as in reinforcement learning).
2. Compute expected behaviors computed from policy π, compared to observed behaviors $\mathcal{B}$, and use to update reward R.

How to use IRL for...

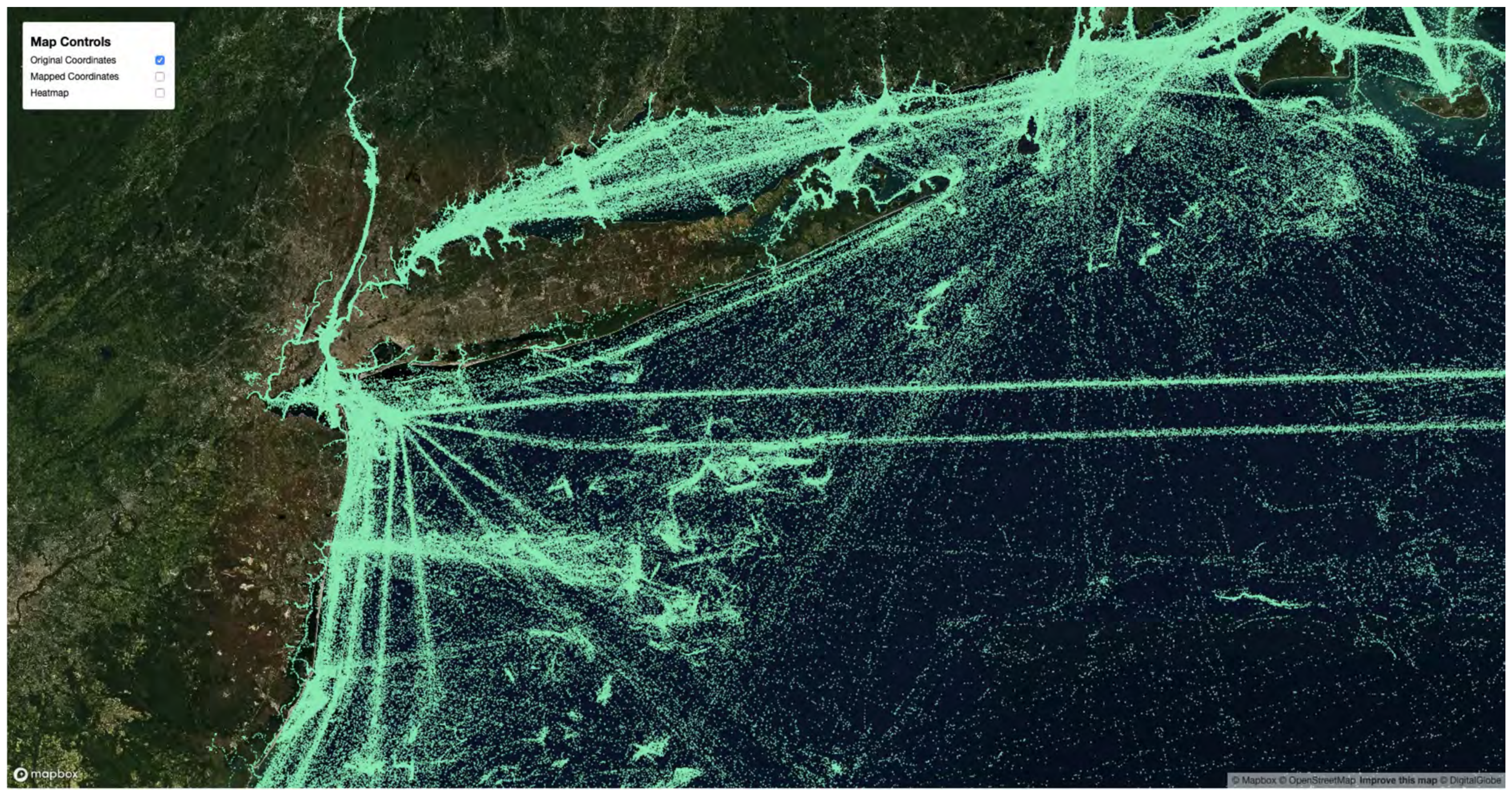
Activity-based intelligence:

1. Learn R from observed behaviors.
2. If new behavior exhibits low-reward actions in states, flag as abnormal.

Teaching expert behavior:

1. Learn R from expert behavior.
2. When a novice is in a state where she doesn't know the proper action, suggest the one with highest reward.

Inverse Reinforcement Learning techniques are an efficient and effective means to perform **activity-based intelligence** or to teach novices how to **perform tasks like experts.**



©Mapbox, ©OpenStreetMap, and ©DigitalGlobe

Goals of this work

1. Apply IRL Techniques to DoD/IC relevant problems:
   - Employ efficient implementations that scale to a large number of observations.
   - Build demonstration data ingestion to visualization tools.
2. Develop Novel IRL Techniques that are robust to rare events.
3. Develop techniques that are able to explain, simulate, and demonstrate expert behavior.

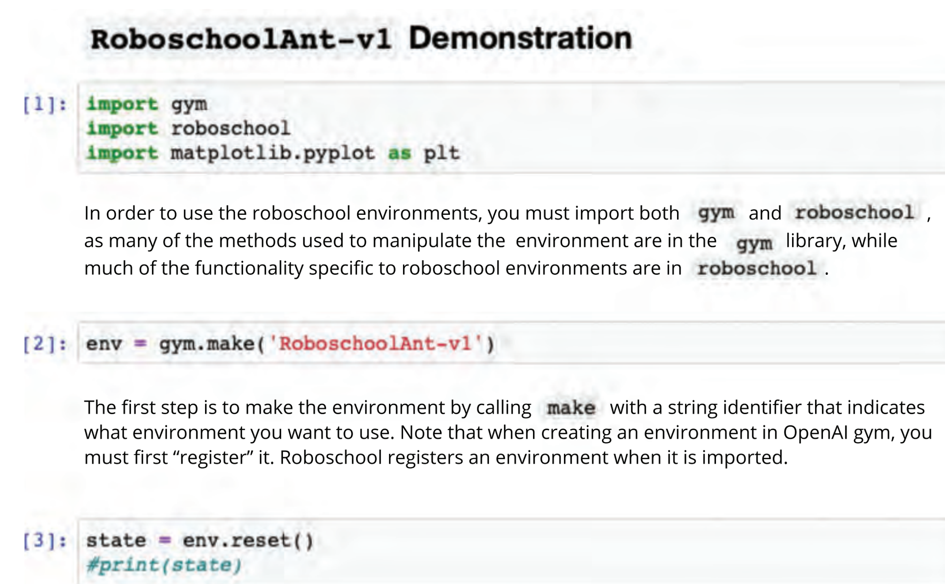
Progress

1. Implementation of Maximum Causal Entropy IRL that is up to **1000x faster** than academic implementation.

   Source: Ziebart, Brian D., J. Andrew Bagnell, and Anind K. Dey. "Modeling interaction via the principle of maximum causal entropy." Proceedings of the 27th International Conference on Machine Learning. Omnipress, 2010.

2. Data ingestion pipeline and visualization of IRL being used to model ship behavior on U.S. Coast Guard Automatic Ship Identification (AIS) data.
3. Investigation into current IRL techniques and how robust they are to rare events, leading to initial formulation of robust IRL technique.
4. Data scientist study to capture expert data scientist behavior.

Additional Figures



RoboschoolAnt-v1 Demonstration

[1]:
```
import gym
import roboschool
import matplotlib.pyplot as plt
```

In order to use the roboschool environments, you must import both `gym` and `roboschool`, as many of the methods used to manipulate the environment are in the `gym` library, while much of the functionality specific to roboschool environments are in `roboschool`.

[2]:
```
env = gym.make('RoboschoolAnt-v1')
```

The first step is to make the environment by calling `make` with a string identifier that indicates what environment you want to use. Note that when creating an environment in OpenAI gym, you must first "register" it. Roboschool registers an environment when it is imported.

[3]:
```
state = env.reset()
#print(state)
```

Carnegie Mellon University
Software Engineering Institute

Eric Heim | etheim@sei.cmu.edu
Developer: Jonathan Hoyle Academic Collaborators: Stephanie Rosenthal (CMU/RI), Anind Dey (UW)