CS 383: ARTIFICIAL INTELLIGENCE

Conclusion and Advanced Applications

Prof. Scott Niekum — UMass Amherst
Overview of AI Topics

Search / Planning
- Uninformed Search
  - A* Search
  - CSPs
  - Local Search
- Minimax
- Expectimax
- MDPs

Machine Learning
- Reinforcement Learning
- Probability Theory
- Bayes Nets
- HMMs
- Particle Filters
- Decision Diagrams

Naive Bayes
- Perceptrons
- Neural Networks
- Kernels
- Clustering
- VPI
Overview of Machine Learning

Supervised Learning
- Discriminative Models
  - Perceptrons
  - Neural Networks
- Generative Models
  - Bayes Nets
  - Naive Bayes
  - HMMs

Reinforcement Learning
- MDPs
- Value Iteration
- Policy Iteration
- Q Learning

Unsupervised Learning
- K-Means Clustering

Supporting Ideas
- Probability Theory
- VPI
- Particle Filters
- Kernels
Maximize Your Expected Utility
Properties of task environment

- Fully observable vs. partially observable
- Single-agent vs. multi-agent
- Deterministic vs. stochastic
- Episodic vs. sequential
- Static vs. dynamic
- Discrete vs. continuous
- Known vs. unknown
Not multi-agent if other agents can be considered part of the environment

Only considered to be multi-agent if the agents are maximizing a performance metric that depends on other agents’ behavior

Single agent example: Pacman with randomly moving ghosts

Multi-agent example: Pacman with ghosts that use a planner to follow him
Single / Multi Agent

**Single**
- Uninformed Search
- A* Search
- Local Search
- CSPs

**Multi**
- Minimax
- Expectimax
- MDPs
- Reinforcement Learning
Deterministic vs. stochastic

- **Deterministic**: next state of environment is completely determined by the current state and the action executed by the agent.
- **Stochastic**: actions have probabilistic outcomes.
- Strongly related to partial observability — most apparent stochasticity results from partial observation of a deterministic system.
- **Example**: Coin flip.
Determinism

Deterministic
- Uninformed Search
- A* Search
- Local Search
- CSPs
- Minimax

Stochastic
- Expectimax
- MDPs
- Reinforcement Learning
- Decision Diagrams
Fully observable vs. partially observable

- Fully observable: agent’s sensors give it access to complete state of the environment at all times
- Can be partially observable due to noisy and inaccurate sensors, or because parts of the state are simply missing from the sensor data
- Example: Perfect GPS vs noisy pose estimation
- Example: IKEA assembly while blindfolded

Almost everything in the real world is partially observable
# Observability

<table>
<thead>
<tr>
<th>Fully Observable</th>
<th>Partially Observable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uninformed Search</td>
<td>POMDPs</td>
</tr>
<tr>
<td>A* Search</td>
<td>Bayes Nets</td>
</tr>
<tr>
<td>Local Search</td>
<td>HMMs</td>
</tr>
<tr>
<td>CSPs</td>
<td>Decision Diagrams</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Known vs. unknown

- Agent’s state of knowledge about the “rules of the game” / “laws of physics”
- Known environment: the outcomes for all actions are given
- Unknown: agent has to learn how it works to make good decisions
- Possible to be partially observable but known (solitaire)
- Possible to be fully observable but unknown (video game)
## Model of the World

### Known

- Uninformed Search
- A* Search
- Local Search
- CSPs
- Minimax
- Expectimax
- MDPs
- Value Iteration
- Decision Diagrams

### Unknown

- Q Learning
- Learning parameters of Bayes Net
Robotic Helicopters
Hover

[Ng et al, 2004]
Autonomous Helicopter Flight

Key challenges:

- Track helicopter position and orientation during flight
- Decide on control inputs to send to helicopter
Autonomous Helicopter Setup

On-board inertial measurement unit (IMU)

Send out controls to helicopter

Position
HMM for Tracking the Helicopter

- **State:**
  \[ s = (x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi}) \]

- **Measurements:** [observation update]
  - 3-D coordinates from vision, 3-axis magnetometer, 3-axis gyro, 3-axis accelerometer

- **Transitions (dynamics):** [time elapse update]
  - \[ s_{t+1} = f(s_t, a_t) + w_t \]
    - \( f \) encodes helicopter dynamics, \( w \): noise
Helicopter MDP

- **State:** $s = (x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi})$

- **Actions (control inputs):**
  - $a_{\text{lon}}$: Main rotor longitudinal cyclic pitch control (affects pitch rate)
  - $a_{\text{lat}}$: Main rotor latitudinal cyclic pitch control (affects roll rate)
  - $a_{\text{coll}}$: Main rotor collective pitch (affects main rotor thrust)
  - $a_{\text{rud}}$: Tail rotor collective pitch (affects tail rotor thrust)

- **Transitions (dynamics):**
  - $s_{t+1} = f(s_t, a_t) + w_t$
  
  [f encodes helicopter dynamics]
  [w is a probabilistic noise model]

- **Can we solve the MDP yet?**
Problem: What’s the Reward?

- Reward for hovering:

\[ R(s) = -\alpha_x (x - x^*)^2 \\
-\alpha_y (y - y^*)^2 \\
-\alpha_z (z - z^*)^2 \\
-\alpha_\dot{x} \dot{x}^2 \\
-\alpha_\dot{y} \dot{y}^2 \\
-\alpha_\dot{z} \dot{z}^2 \]
Problem: What’s the Reward?

- Rewards for “Flip”?
  - Problem: what’s the target trajectory?
  - Just write it down by hand?
  - Penalize for deviation from trajectory
Flips (?)
Helicopter Apprenticeship?
Learning task objectives: Inverse reinforcement learning

Reinforcement learning basics:

MDP: \((S, A, T, \gamma, D, R)\)

Policy: \(\pi(s, a) \rightarrow [0, 1]\)

Value function: \(V^\pi(s_0) = \sum_{t=0}^{\infty} \gamma^t R(s_t)\)

What if we have an MDP/R?
Learning task objectives: Inverse reinforcement learning

1. Collect user demonstration \((s_0, a_0), (s_1, a_1), \ldots, (s_n, a_n)\)
   and assume it is sampled from the expert’s policy, \(\pi^E\)

2. Explain expert demos by finding \(R^*\) such that:

\[
E[\sum_{t=0}^{\infty} \gamma^t R^*(s_t)|\pi^E] \geq E[\sum_{t=0}^{\infty} \gamma^t R^*(s_t)|\pi] \quad \forall \pi
\]

\[
E_{s_0 \sim D}[V^E^\pi(s_0)] \geq E_{s_0 \sim D}[V^\pi(s_0)] \quad \forall \pi
\]

How can search be made tractable?

[Abbeel and Ng 2004]
Learning task objectives: Inverse reinforcement learning

Define $R^*$ as a linear combination of features:

$$R^*(s) = w^T \phi(s), \text{ where } \phi : S \to \mathbb{R}^n$$

Then,

$$E[\sum_{t=0}^{\infty} \gamma^t R^*(s_t)|\pi] = E[\sum_{t=0}^{\infty} \gamma^t w^T \phi(s_t)|\pi]$$

$$= w^T E[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)|\pi]$$

$$= w^T \mu(\pi)$$

Thus, the expected value of a policy can be expressed as a weighted sum of the expected features $\mu(\pi)$

[Abbeel and Ng 2004]
Learning task objectives: Inverse reinforcement learning

Originally - Explain expert demos by finding $R^*$ such that:

$$
E\left[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) | \pi^E\right] \geq E\left[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) | \pi\right] \ \forall \pi
$$

Use expected features:

$$
E\left[\sum_{t=0}^{\infty} \gamma^t R^*(s_t) | \pi\right] = w^T \mu(\pi)
$$

Restated - find $w^*$ such that:

$$
w^* \mu(\pi^E) \geq w^* \mu(\pi) \ \forall \pi
$$

[Abbeel and Ng 2004]
Learning task objectives: Inverse reinforcement learning

**Goal:** Find $w^*$ such that: $w^* \mu(E) \geq w^* \mu(\pi) \ \forall \pi$

1. Initialize $\pi_0$ to any policy

Iterate for $i = 1, 2, \ldots$:

2. Find $w^*$ s.t. expert maximally outperforms all previously examined policies $\pi_0 \ldots i-1$:

$$\max_{\epsilon, w^*: \|w^*\|_2 \leq 1} \epsilon \text{ s.t. } w^* \mu(E) \geq w^* \mu(\pi_j) + \epsilon$$

3. Use RL to calc. optimal policy $\pi_i$ associated with $w^*$

4. Stop if $\epsilon \leq$ threshold

[Abbeel and Ng 2004]
Learning task objectives: Inverse reinforcement learning

Goal: Find $w^*$ such that: $w^* \mu(\pi^E) \geq w^* \mu(\pi) \, \forall \pi$

1. Initialize $\pi_0$ to any policy

Iterate for $i = 1, 2, \ldots$:

2. Find $w^*$ s.t. expert maximally outperforms all previously examined policies $\pi_0, \ldots, \pi_{i-1}$:

   $$\max_{\epsilon, w^*: \|w^*\|_2 \leq 1} \epsilon \quad \text{s.t.} \quad w^* \mu(\pi^E) \geq w^* \mu(\pi_j) + \epsilon$$

3. Use RL to calc. optimal policy $\pi_i$ associated with $w^*$

4. Stop if $\epsilon \leq$ threshold

[Abbeel and Ng 2004]
Quadruped

- Low-level control problem: moving a foot into a new location \( \rightarrow \) search with successor function \( \sim \) moving the motors

- High-level control problem: where should we place the feet?
  - Reward function \( R(x) = w \cdot f(s) \) [25 features]

[Kolter, Abbeel & Ng, 2008]
Experimental setup

- Demonstrate path across the “training terrain”
- Run apprenticeship to learn the reward function
- Receive “testing terrain”---height map.
- Find the optimal policy with respect to the learned reward function for crossing the testing terrain.

[Kolter, Abbeel & Ng, 2008]
Without learning
With learned reward function
Problems with standard inverse reinforcement learning

Policy learning in inner loop

IRL Loop

Reward update

Policy learning

Cannot outperform demonstrator

Argh!
T-REX: Trajectory-ranked Reward Extrapolation

- Fully supervised — no policy learning
- Works on high-dim (e.g. Atari) with ~10 demos
- Auto-generated rankings: 

\[ L(\theta) = E_{\tau_i, \tau_j \sim \Pi} \left[ \xi \left( \frac{\exp \sum_{s \in \tau_i} \hat{r}_\theta(s)}{\exp \sum_{s \in \tau_i} \hat{r}_\theta(s) + \exp \sum_{s \in \tau_j} \hat{r}_\theta(s)} \right) \right] \]

D.S. Brown, W. Goo, P. Nagarajan, and S. Niekum.
Extrapolating Beyond Suboptimal Demonstrations via Inverse Reinforcement Learning from Observations.

D. Brown, W. Goo, and S. Niekum.
Ranking-Based Reward Extrapolation without Rankings
Conference on Robot Learning (CoRL), October 2019.
Ranked demonstrations: HalfCheetah

12.52

44.98

88.97
Results: HalfCheetah

Best demo (88.97)  T-REX (143.40)
Results: Atari

Best demo (600)

T-REX (1495)
Frame stacks: best vs. worst reward

Worst

Best
D-REX: Auto-generated rankings

Unranked Demonstrations  \( \Rightarrow \pi_{BC} \Rightarrow \) Noise-modified Rollouts  “Ranked” Trajectories

Multimodal data sources: Human Gaze
Collecting gaze data during demonstrations

Tobii 2 Glasses

Image-based gaze tracking

A. Saran, S. Majumdar, E.S. Short, A.L. Thomaz, and S. Niekum.
Human Gaze Following for Human-Robot Interaction.
Human gaze during ambiguous task demonstrations

Gaze fixations during kinesthetic demonstration

A. Saran, E.S. Short, A.L. Thomaz, and S. Niekum.
Understanding Teacher Gaze Patterns for Robot Learning.
Conference on Robot Learning (CoRL), October 2019.
CGL: Coverage-based Gaze Loss

A. Saran, R. Zhang, E.S. Short, and S. Niekum.
Efficiently Guiding Imitation Learning Algorithms with Human Gaze.
Multimodal data sources: **Facial Reactions**

Implicit human feedback:

- Occurs naturally
- Is not necessarily intended to influence behavior
- Can be used with no additional burden on user
EMPATHIC: Learning from implicit feedback — training

Y. Cui, Q. Zhang, A. Allievi, P. Stone, S. Niekum, and W. Knox.
The EMPATHIC Framework for Task Learning from Implicit Human Feedback.
Conference on Robot Learning (CoRL), November 2020.
EMPATHIC: Learning from implicit feedback — deployment

Y. Cui, Q. Zhang, A. Allievi, P. Stone, S. Niekum, and W. Knox.
The EMPATHIC Framework for Task Learning from Implicit Human Feedback.
Conference on Robot Learning (CoRL), November 2020.
Even more multimodal data sources

**Auxiliary video alignment**

**Natural language**

“Rotate the red handle downward”

**Audio and prosody**


A. Saran and S. Niekum. Analyzing Audio Patterns During Demonstration In Submission.
Demonstration

[RSS 2013, IJRR 2015]
High-level task modeling

Why?
- Superior generalization of skills
- Handle contingencies
- Adaptively sequence skills

Questions
- How many skills?
- Parameters of skills / controllers?
- How to sequence intelligently?

Unsegmented demonstrations of multi-step tasks

Finite-state task representation
System overview
System overview
System overview

Joint angles → Forward kinematics → Task demos
Gripper pose → Object recognition → Stereo data
System overview

Joint angles → Task demos → Preprocessing / BP-AHMM segmentation → Segmented motions

Gripper pose → Forward kinematics → Object recognition

Stereo data

[IROS 2012]
Segmenting demonstrations

Motion categories

Observations

Standard Hidden Markov Model
Segmenting demonstrations

\[ y_t^{(i)} = \sum_{j=1}^{r} A_{j,z_t^{(i)}} y_{t-j}^{(i)} + e_t^{(i)}(z_t^{(i)}) \]

Motion categories

Observations

Autoregressive Hidden Markov Model
Segmenting demonstrations

\[ y_t^{(i)} = \sum_{j=1}^{r} A_{j,z_t^{(i)}} y_{t-j}^{(i)} + e_{t}^{(i)} (z_{t}^{(i)}) \]

Motion categories

Observations

Autoregressive Hidden Markov Model

[IIOS 2012]
Segmenting demonstrations

\[ y_t^{(i)} = \sum_{j=1}^{r} A_{j,z_t^{(i)}} y_{t-j}^{(i)} + e_t^{(i)} (z_t^{(i)}) \]
Segmenting demonstrations

\[ y_t^{(i)} = \sum_{j=1}^{r} A_{j,z_t^{(i)}} y_{t-j}^{(i)} + e_t^{(i)} (z_t^{(i)}) \]

Motion categories

Observations

Beta Process  Autoregressive Hidden Markov Model
(Fox et al. 2011)

[ IROS 2012 ]
System overview

Task demos
Forward kinematics
Joint angles
Object recognition
Stereo data
Preprocessing / BP-AR-HMM segmentation
Segmented motions
Coordinate frame detection
Frame-labeled segments
Gripper pose
Stereo data
Red object coord. frame

[IROS 2012]
System overview

Joint angles
Gripper pose
Stereo data
Task demos
Forward kinematics
Object recognition
Preprocessing / BP-AR-HMM segmentation
Segmented motions
Coordinate frame detection
Frame-labeled segments
Learning from demonstration
DMPs with frame-relative goals

[IROS 2012]
**System overview**

- **Task demos**
- **Joint angles**
- **Gripper pose**
- **Stereo data**

**Forward kinematics**

- **Preprocessing /**
  - **BP-AHMM**
  - **segmentation**

**Segmented motions**

- **Coordinate frame**
  - **detection**

**Frame-labeled segments**

- **Learning from demonstration**
  - **DMPs**
  - **with frame-relative goals**

- **Red object coord. frame**

**Frame-labeled segments**

- **Frame-labeled segments**

**Learning from demonstration**

- **DMPs**
  - **with frame-relative goals**

**Realtime data from a novel task**

- **Forward kinematics**
- **Joint angles**
- **Gripper pose**
- **Object recognition**
- **Stereo data**

[IROS 2012]
System overview
Learning a task plan: Finite state automata
Learning a task plan: Finite state automata

Controller built from motion category examples

Classifier built from robot percepts

[RSS 2013, IJRR 2015]
Interactive corrections
Replay with corrections: missed grasp
Replay with corrections: too far away
Replay with corrections: full run