Learning User Models with Limited Reinforcement: An Adaptive Human-Robot Interaction System

Finale Doshi
Nick Roy
Motivation
(our goals, how you can help)
Motivation

• Wouldn't it be nice if we could simply tell a robot to go to a particular location? Follow along side someone?

• This ability would be particularly useful to wheelchair users with severely limited mobility.
The Problem

Speaking to robots involves several challenges:

- Noisy speech recognition
- Linguistic ambiguities
  - multiple “elevators” may exist
  - the robot must know that an “elevator” is a location
  - other ambiguous phrases
The planning process

Robot

- observation
- state estimation
- sensors

User

- action
- observed
- hidden
- user dynamics

Dialog model

Belief about the user

State of the user
The planning process

Robot

- observation
- state estimation
- dialog model
- planning

User

- sensors
- State of the user
- user dynamics

I want to go to the elevator

Belief about the user

user
hidden

observed

LangRo 2007
The planning process

Robot

observation

sensors

User

dialog model

Belief about the user

I want to go to the elevator.

I want to go to the elevator.
The planning process

Robot

User

Maybe he wants the elevator, but I'm not sure

state estimation

dialog model

Belief about the user

planning

observation

sensors

State of the user

action

observed hidden

user dynamics

I want to go to the elevator.

I want to go to the elevator.

waiter, mumble, elevator...
The planning process

**Robot**

- waiter, mumble, elevator...
- observation
- state estimation

**User**

- I want to go to the elevator.
- sensors
- State of the user
- user dynamics

**Dialog Model**

- Belief about the user
- planning

- Do you want to go to the elevator?
- observed hidden
Partially Observable Markov Decision Process

- States (hidden!): the user's wants
- Observations: what the robot hears
- Actions: movements and queries the robot can do
- Reward Model $R(s,a)$
- Transition Model $T(s'|s,a)$
- Observation Model $O(o|s,a)$
Currently we use a bag of words approach with keyword list.

- Each location has a keyword (e.g., 'cafe' for cafe)
- Additional keywords also included (e.g., 'food', 'tea', or 'lunch')

Model is probability of each observation in each state.

Could be adjusted to handle more sophisticated language models.
Why is the POMDP model useful?

- We track our belief, a probability distribution over states.
- We choose an action based on our belief, thus taking into account our uncertainty about what the user really wants.
- POMDPs have been used in several dialog management applications, such as Roy, Pineau, and Thrun (2000) and Williams and Young (2005).
Why is the POMDP model useful?

- We track our *belief*, a probability distribution over states.
- We choose an action based on our belief, thus taking into account our uncertainty about what the user really wants.

![Diagram showing probability distribution for action choices: cafe, elevator, car. The best action is clarifying where to go.]

- POMDPs have been used in several dialog management applications, such as Roy, Pineau, and Thrun (2000) and Williams and Young (2005)
Difficulties with the POMDP model

Even a 5-state model has 1344 parameters!

We would like to learn these parameters online.
Algorithm
(learning the user model)
How can we learn the model online?

Ignore uncertainty: **fast, not robust**

Plan with parameters as hidden state: **robust but slow**
How can we learn the model online?

- Ignore uncertainty: fast, not robust
- Approximate planning with Bayes risk, meta-queries
- Plan with parameters as hidden state: robust but slow
Approximate Planning Strategy

Idea: approximate risk of each action; if risk is large, ask for help

Algorithm:

1. Prior over goal, models
2. What is the Bayes risk of safest action?
3. Do safest action.
4. Ask for help (meta-query)
5. Risk okay or too high?
6. Update belief over goal, models as appropriate
7. Get an observation

Diagram:
- Prior over goal, models
- What is the Bayes risk of safest action?
- Do safest action.
- Ask for help (meta-query)
- Update belief over goal, models as appropriate
- Get an observation
Approximate Planning Strategy

Idea: approximate risk of each action; if risk is large, ask for help

Algorithm:

1. **Prior over goal, models**
2. **What is the Bayes risk of safest action?**
   - **Risk okay**
     - **Do safest action.**
   - **Risk too high**
     - **Ask for help (meta-query)**
3. **Posterior over goal, models**
4. **Update belief over goal, models as appropriate**
5. **Get an observation**
Action Selection with Bayes Risk

- Find the action with the minimal risk:

$$a = \arg\min_{a \in A} \int_{M} (Q_m(b_m, a) - Q_m(b_m, a_m')) p(m) \, dm$$

If the risk is more than the cost of a meta-query, ask for help.

- Evaluate the Bayes Risk integral approximately using sampled POMDPs:

$$a = \arg\min_{a \in A} \sum_{i} (Q_i(b_i, a) - Q_i(b_i, a_i')) w_i$$
Approximate Planning Strategy

Idea: approximate risk of each action; if risk is large, ask for help

Algorithm:

- Prior over goal, models
- What is the Bayes risk of safest action?
- Risk okay? Do safest action.
- Risk too high? Ask for help (meta-query)
- Posterior over goal, models
- Update belief over goal, models as appropriate
- Get an observation

LangRo 2007
Meta-Queries

Idea: give the user a sense of the robot's uncertainty so that he can provide feedback to help discover the model.

Benefits:

- Robot does not need to take large risks to determine that a particular decision may be poor.
- User only needs to provide reinforcement when the robot is sufficiently confused.
Meta-Queries

Questions of the form:

- “I think you might want to go to the printer. Should I go to the printer?”
- “I'm certain you want to go to the printer. Should I go to the printer?”
- “Instead, should I ask for you to confirm your location?”

Ask these questions until we determine the correct action.
Approximate Planning Strategy

Idea: approximate risk of each action; if risk is large, ask for help

Algorithm:

1. Prior over goal, models

2. What is the Bayes risk of safest action?
   - Risk okay
   - Risk too high

3. Do safest action.

4. Ask for help (meta-query)

5. Update belief over goals, models as appropriate

6. Posterior over goal, models

7. Get an observation
How to Sample POMDPs: Belief Update

Two sources of information affect our belief over models:

- **h**: most recent history $a_1 o_1 a_2 o_2 ... a_n o_n$
- **Q**: query-response info $\{ (q_k, r_k, h_k) \}$

Bayes Rule

$$p(m|h, Q) \propto ...$$

$$p(Q|m) p(h|m) p(m)$$

Note: we resample when update is likely to be accurate (e.g., at the completion of a dialog).
If we begin with a Dirichlet prior, the history information can be incorporated in closed form.

We sample from the this updated distribution over models, throwing out samples that violate too many meta-queries.

(Note: we require rejection sampling because we cannot compute how the Dirichlet is truncated.)
Results
Simulation Results: Full Model Learning

Reasonable Prior

Non-informative Prior
User Test Results: Example Dialog 1

Early Conversation:
User: Give me the forecast.
Robot: I'm confused. What action should I take now?
<User indicates that the robot should provide the weather forecast>
Robot: Showers

Later Conversation:
User: What's the forecast for today?
Robot: Do you want the weather?
User: Yup.
Robot: Showers
User Test Results: Example Dialog 2

**Early Conversation:**
User: I want some tea from the kitchen.
Robot: How can I help you?
User: The tea is in the kitchen near the coffee machine. *The system hears "copy" instead of "coffee" and is confused about "tea" and "kitchen."*
Robot: How can I help you?
User: Go to the kitchen.
Robot: I'm confused. What action should I take now?
*The user indicates "Confirm the coffee machine area" is the correct action.*
Robot: Do you want to go to the coffee machine area?
User: Yup.
...
*The system asks a few more (non-meta) questions*
Robot: Do you want to go to the coffee machine area?
User: Yup.
Robot: Going to the coffee machine area.

**Later Conversation:**
User: Take me to the kitchen. *The system hears "Nick" and "kitchen."*
Robot: How can I help you?
User: I want some coffee. *The system hears "copy" instead of "coffee."*
Robot: Do you want to go to the coffee machine area?
User: Yup.
Robot: Going to the coffee machine area.
Conclusions and Future Work

● Although POMDP dialog models require many parameters, we can learn those parameters online.
  − Bayes risk action selection lets the robot act robustly in the face of model uncertainty
  − The learning process can be further improved by incorporating meta-queries.

● Extensions
  − Improve efficiency (better sampling of POMDPs)
  − More intuitive meta-query mechanisms
  − Coupling with more sophisticated language models
Thank-you!
Discrete Models: Why few policies?

In the special case where:

- Only rewards are unknown
- Simple dialog model

The *policies* for a variety of parameter values are similar; the main degree of freedom is how certain we must be before acting, which translates to how many times to confirm a choice.
Error in Approximating Bayes Risk

• If we want to estimate if the Bayes Risk is greater than \( \zeta \) with confidence \( \delta \), two error sources exist:
  
  – Error due to approximating risk from samples:
    \[
    n_m = \frac{(R_{\text{max}} - \min(\zeta, R_{\text{min}}))^2}{2(1 - \gamma)^2 \epsilon_m^2} \log \frac{1}{\delta}
    \]

  – Error due to approximate POMDP solutions:
    \[
    \epsilon_{pb} = 2 \delta_b \frac{(R_{\text{max}} - R_{\text{min}})}{(1 - \gamma)^2}
    \]

• Noting that \( \zeta = \epsilon_m + \epsilon_{pb} \), set \( \epsilon_m \) and \( \epsilon_{pb} \) to trade between the number of belief samples and model samples.
Belief Update

- If we begin with a Dirichlet prior, the history information can be incorporated in closed form.

- We sample from the this updated distribution over models, throwing out samples that violate too many meta-queries.
Belief Update: History Information

- Use history information to analytically update Dirichlet prior over models:

\[ p(m|h, Q) = \eta p(Q|m)p(h|m)p(m) \]

- Dirichlet update requires state history; estimate the state sequence using the standard forward-backward algorithm.

- EM-like update; will converge to some local optimum.
Belief Update: Query information

• Next use importance sampling to sample POMDPs; choose a $p(Q|m)$ that forgives noise in the policy query response and the approximate POMDP solution.

$$p(m|h, Q) = \eta \frac{1}{1+k} u(k' - k)$$

$$p(m|h, Q) = \eta \frac{1}{1+k} u(k' - k) p(Q|m) p(h|m) p(m)$$

• For a real time system, apply additional heuristics:
  - Try sampling until the original minimum error is reduced
  - Try a convex combination of new and good samples
  - Limit number of samples to try
Continuous Models: Belief Update

- If **only** the reward model is unknown, we can efficiently prune the reward space:

  ![Diagram of reward space pruning]

  - We can use rejection sampling or MC MC techniques to sample from valid regions in the reward space.
Performance Guarantees

- We can provide a lower bound on the expected performance of our approach compared to the optimal policy:

$$V' > \eta \left( V - \frac{\xi}{(1-\gamma)} \right) + (1-\eta) \left( \frac{R_{\text{min}}}{(1-\gamma)} \right), \quad \eta = \frac{(1-\gamma)(1-\delta)}{1-\gamma(1-\delta)}$$

- Since the Dirichlet counts are always increasing, we will eventually converge to some transition and observation model.
Termination Bounds

- To estimate if the probability of asking a meta-query after n more interactions is greater than $\zeta$ with confidence $\delta$, we can:
  - Compute “worst posterior” by assigning interaction counts to make a flat Dirichlet posterior.
  - Sample POMDPs from the posterior.
  - Sample beliefs from the POMDPs.
  - Reject if $f(\zeta)$-proportion beliefs require meta-queries.

- We can set the number of POMDP, belief samples required, as well as $f(\zeta)$, based on our desired confidence.
Solving the POMDP Dialog Model

Value of a belief

$$V_n(b) = \max_a Q_n(b, a)$$

Value of belief, action pair

$$Q_n(b, a) = R(b, a) + \gamma \sum_{b' \in B} T(b' | b, a) V_{n-1}(b')$$

$$Q_n(b, a) = R(b, a) + \gamma \sum_{o \in O} O(o | b, a) V_{n-1}(b^o)$$

Current reward

Future Reward
Dialog Model: Solving the POMDP

We think of the previous recursions as building a policy tree...

Action: Go to elevator

Certain that we're in state one: User wants to go to the elevator

Certain in state two: User wants to go to the cafeteria
Dialog Model: Solving the POMDP

We think of the previous recursions as building a policy tree...

Action: Ask for location

<table>
<thead>
<tr>
<th>State one: User wants to go to the elevator</th>
<th>State two: User wants to go to the cafeteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expected Reward</td>
<td>Expected Reward</td>
</tr>
<tr>
<td>-500</td>
<td>+100</td>
</tr>
</tbody>
</table>
Dialog Model: Solving the POMDP

We think of the previous recursions as building a policy tree; planning ahead increases our expected reward.

Action: Ask for location
Observe “elevator”
Observe “cafeteria”
Action: Go to elevator
Action: Go to cafeteria
Expected Reward
State one: User wants to go to the elevator
State two: User wants to go to the cafeteria

+100
-500
Dialog Model: Solving the POMDP

Given multiple trees, we can determine the most appropriate action:

Action: Go to elevator
Action: Ask for location
Observe "elevator"
Action: Go to elevator
Action: Go to cafeteria
Observe "cafeteria"
Action: Go to cafeteria

Expected Reward

State one: User wants to go to the elevator
State two: User wants to go to the cafeteria

+100
-500
1. Modeling Uncertainty: Placing Priors

- An expert provides guesses using “pre-observation counts.” For example: suppose I think $P(\text{error}) = 10\%$
  - If I'm pretty certain, guess “saw 100 errors in 1000 tests”
  - If I'm not sure, might guess “saw 1 error in 10 tests”

- Parameter uncertainty induces uncertainty in the value function.
Prior Work

- POMDP Dialog Management: (assume model known)
  - Roy et. al.: nursing home aide
  - Williams and Young: automated telephone operator

- Bayesian (PO)MDP Model Learning:
  - Dearden et. al.: Bayesian MDP model learning
  - Beetle (Poupart et. al.): frame unknown MDP as a continuous state POMDP
  - Medusa (Jaulmes et. al.): sample from a distribution over POMDPs; use the sample for action selection
Meta-Queries (Discrete Models)

- Choose sets of parameters that produce different policies; let each of these be a user preference model.
- Design meta-action queries to differentiate between the models.
- Solve just like the parameter POMDP.

Effect of Meta-Actions on Total Reward, Harsh User

<table>
<thead>
<tr>
<th>Without meta-actions</th>
<th>With meta-actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Reward</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

LangRo 2007
BR Action Selection (Continuous Models)

Observation Learning

Reward Learning
Dialogs with Meta-Queries

- Meta-queries decrease with experience (from user trials, significant at the 5% level)

<table>
<thead>
<tr>
<th>Times a place requested</th>
<th>First</th>
<th>Second</th>
<th>Third</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total number of requests</td>
<td>29</td>
<td>15</td>
<td>8</td>
</tr>
<tr>
<td>Requests with meta-queries</td>
<td>22</td>
<td>11</td>
<td>2</td>
</tr>
<tr>
<td>Proportion with meta-queries</td>
<td>0.76</td>
<td>0.73</td>
<td>0.25</td>
</tr>
</tbody>
</table>