Learning in dialogue systems

What is learning?

- From Mitchell, 1997:

  "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P improves with experience E."

What do you need to learn?

- Task T
  - What's the task in a dialogue system?
- Experience E
  - What data can we use to learn?
  - How are these data annotated?
- Performance measure P
  - How do we know we're learning?
  - How do we measure success?

Where has learning been applied?

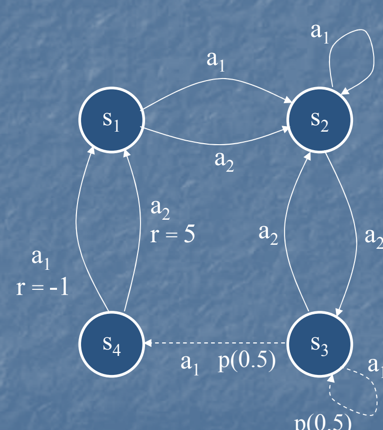
- Medical diagnosis
- Credit card fraud
- Stock market analysis
- Game playing

Reinforcement learning

- Learning is associated with a *reward*
- By optimizing reward, algorithm learns optimal strategy
- Key assumption: problem can be divided into states, transitions, and actions associated with those transitions (you have to have something to associate reward with!)
- A Markov Decision Process!

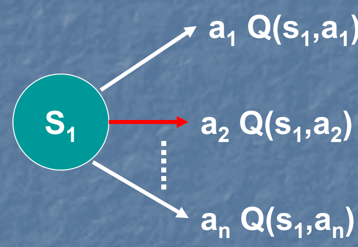
Markov Decision Processes

- State, s
- Actions, a
- Rewards, r
- Transitions (associated with actions)
- Formalizes problem—when in state S, what is the utility (reward) for taking a particular action, among the choices of actions possible?

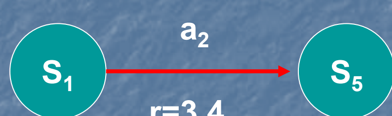


Q Learning (Watkins, 1989)

- Every action has a Q value associated with it
- Actions are explored according to a policy or at random

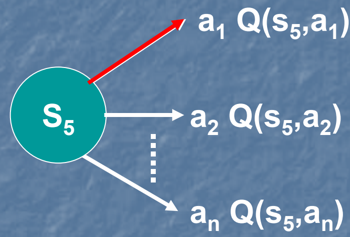


Rewards



- Once we have taken a transition, we receive a reward

Moving ahead



- At this point, we measure the utility of the state by maximizing the Q value
- Once new state is chosen, we update Q value of preceding state according to reward and new state's Q value

Problems with Q learning/RL in general

- State space can be huge, therefore
  - Time to search it can be quite long
  - Memory requirements can be quite big
  - States can be missed in search
- The "curse of dimensionality"

Today's question

- How is a SDS system like a Markov Decision Process?
- What can be learned automatically in a spoken dialogue system?
- What's the "experience"—what data can be used to learn?
- What performance evaluation can be optimized in learning algorithm?

Applying RL to Spoken Dialogue Systems (Levin and Pieraccini, '97)

- Consider a simple form-filling task: booking a flight
- Constraints that need to be elicited form user:
  - Source
  - Destination
  - Date
  - Time
  - Airline
- Problem: How do determine (automatically) the optimum order of elicitation for these constraints?

Example Air Travel Application

- User: I'd like to fly to Hong Kong.
- System1: There are 539 flights every day to Hong Kong. British Airways flight 63 leaves at 8:00 a.m. from London Heathrow. Air France flight 48 leaves at 8:05 a.m. from Paris Charles de Gaulle ….
- System2: There are 539 flights every day to Hong Kong. What time will you be flying?
- System3: Where are you leaving from?

Response strategies

- Go to the database, retrieve all flights that match user constraints, read them off.
- Go to the database, summarize contents, select a constraint at random and ask user to indicate a preference.
- Ask constraints in a predetermined order.

What to optimize

- Speed – database retrieval
- User frustration – long responses
- "Reasonableness"

State-Based Dialogue Manager

- System **state**:
  - attributes perceived so far
  - other dialogue history info
  - data on particular user
- Dialogue **strategy**: mapping from current state to system **action**
- Typically hundreds of states, several reasonable actions from each state

State-Based Dialogue Manager

- Example system states:
  - [dest: NYC; src: LHR; date: ___; time ___; airline: __]
- Example dialogue strategies:
  - Go to db, read list of flights from NYC to LHR
  - Ask user for date of travel OR time OR airline

The Markov assumption applied to dialogue systems

$$\Pr(s_{t+1} \mid s_t, s_{t-1} \ldots s_0, a_t, a_{t-1} \ldots a_0)$$
$$= \Pr(s_{t+1} \mid s_t, a_t)$$

- Probability of transition to new state (S_{t+1}) dependent on preceding state/action
- This is just the start—we still need reward/costs to do reinforcement learning
- Any time an action is chosen, the system incurs a cost
- Final cost is the sum of all costs over particular dialogue strategy

How do assign costs/reward?

- On utterance-by-utterance basis
  - Partial info from user may result in large output from DB query
  - Slow and irritating delivery of results
  - Can be done with user simulation
- As function of overall dialogue
  - Task completion (Did user get information?)
  - Time to completion (Longer=worse)
  - User satisfaction (Was user happy with interaction?)
  - Typically requires real user data

Estimating costs

$$C = W_i \langle N_i \rangle + W_r \langle N_r \rangle + W_o \langle f_o(N_o) \rangle + W_s \langle F_s \rangle$$

- N_i = expected length of interaction in number of turns
- N_r = expected number of tuples returned by the database (cost of retrieving information)
- $f_o(N_o)$ = data presentation cost
- F_s = overall task success

Defining optimal value

$$V^*(s) = \left\langle \sum_{t=0}^{T_F} c(s_t, a_t) \right\rangle$$

- V*(s) = optimal value of state s
- Computed as expected sum of costs from state s, following optimal strategy until final state s_F is reached

Computing optimal value function

$$V^*(s_t) =$$
$$\min_a [\langle c(s_t, a) \rangle] + \sum_{sb} \Pr(s_{t+1} = s \mid s_t, a) V^*(s)]$$

- Optimal value of state s_t is the sum of the instantaneous cost plus the expected value of the next state, using the best available action

Assigning overall utility

- Best path through network is the one that maximizes utility function, which is itself dependent on reward function
- Therefore, best dialogue strategy is the one that receives highest utility

Performing the experiment

- In flight domain:
  - Simulate user interaction, with costs assigned to db retrieval, reward associated with getting flight

Using resulting data

- With resulting dialogues, compute costs/rewards
- Using state network, assign costs to transitions
- Use costs to determine how to move forward in network
- *Use resulting network on real users to determine if dialogue strategy works*

What do we learn about dialogue systems?

- Don't greet user and then hang up
- Don't present a list of flights after only eliciting destination
- Best constraint-elicitation order: source, destination, airline, date, time

Applying RL to Spoken Dialogue Systems (Walker, '00)

- Consider a system providing access to email over the phone
- Dialogue strategies:
  - Summarize email by sender
  - Summarize email by subject
  - Summarize email by both subject and sender
- Problem: What is the best summarization method?

Sample Elvis Dialogue

- *A: Hi, Elvis here. Who is that?*
- *U: Susan*
- *A: Hi Susan, Please wait. I'll get your mail….. Okay, I've got your mail.*
- *U: Summarize my messages.*
- *A': In your top level inbox, you are reading both old and new messages. There's 1 message from Michael , two messages from Kim, one message from Lee, and two messages from Dana.* **(SUMMARIZESYSTEM)**
- *U: Read me my messages from Kim.*
- *A: In the messages from Kim, there's 1 message about Interviewing Antonio and 1 message about Meeting Today . The first message is titled "Interviewing Antonio". It says 'I'd like to interview him. I could also go along to lunch. Kim'* **(READ FIRST STRATEGY)**

Elvis cont.

- *U: Next Message.*
- *A: This is a message about Meeting Today. The meeting this morning is at 10:00 in 2d516. See you there. Kim.*
- *U: Repeat.*
- *A: This is a message about Meeting Today. The meeting this morning is at 10:00 in 2d516. See you there. Kim.*
- *U: I'm done here.*
- *A: You are back to your toplevel inbox.*
- *U: Bye Bye Elvis.*
- *A: Do you want to hang up now?*
- *U: Yes*
- *A: Bye and don't be lonely.*

ELVIS READ STRATEGIES

- READFIRST
- *In the messages from Kim, there's 1 message about Interviewing Antonio and 1 message about Meeting Today. The first message is titled "Interviewing Antonio". It says......*
- READSUMMARYONLY
- *In the messages from Kim, there's 1 message about Interviewing Antonio and 1 message about Meeting Today.*
- READCHOICEPROMPT
- *In the messages from Kim, there's one message about Interviewing Antonio and 1 message about Meeting Today. To hear the messages, say 'Interviewing Antonio' or 'Meeting'.*

ELVIS SUMMARIZE STRATEGIES

- SUMMARIZESYSTEM
- SUMMARIZECHOICE PROMPT
- SUMMARIZEBOTH

- *In your top level inbox, there's one message about fax, one message about Interviewing Antonio, one message about Meeting today, one message about Where I'll Be; and one message about Evaluation Technical Memo*
- *Summarize by Subject, by Sender or Both?.*
- *In your top level inbox, from Dana, there's one message about fax; from Kim, there's one message about Interviewing Antonio and one message about Meeting today; from Lee, there's one message about Where I'll Be; and from Michael, there's one message about Evaluation Technical Memo.*

System Initiative Strategy

- *A: Hi, Elvis here. You have 5 new and no unread messages in your inbox. Say Read, Summarize, or say Help for more options.*
- *U: Read.*
- *A: Select by Sender, Subject or in order of arrival?*
- *U: Sender*
- *A: Which Sender?*
- *U: Kim.*
- *A: In the messages from Kim, there's 1 message about interviewing Antonio and 1 message about.....*

Mixed Initiative Strategy

- *A: Hi, Elvis here. I've got your mail.*
- *U: Read my messages from Kim.*
- *A: In the messages from Kim, there's 1 message about Interviewing Antonio and 1 message about Meeting Today……..*

What should be optimized?

- Task Completion Rates
- Efficiency Issues: Elapsed Time, Dialogue Length
- Qualitative Issues:
  - Do you have to repeat yourself?
  - Does the system misunderstand you?
  - Do you know what to do at each point?
  - Is the system flexible enough?
- Speech Recognizer Performance
- User Expertise

State-Based Dialogue Manager

- Example system states:
  - [user: MW; num_msgs: 5]
- Example dialogue strategies
  - Summarize all messages by sender and subject
  - Summarize all messages by sender’s name only

Performing the experiment

- Get users to interact with various systems, configured for different types of interaction
- Measure time to completion
- Ask users specifically how they felt about interaction

ELVIS EXPERIMENTS

- ELVIS w. initiative/presentation strategies
- ~1M global policies explored
- Instructions to users on web pages
- Training: 78 users each do 3 tasks
- Testing: 6 users do 3 tasks
- Automatic logging of metrics, states, acts
- Users specify task solution on web page
- User Satisfaction survey per dialog

ELVIS State Space for RL

- Know User Name: 0,1
- Initiative Strategy: SI, MI
- Task Progress: 0,1,2
- UserGoal: 0,R,S
- RL state space is 18 states, 2 choices in 1 state, 3 choices in 12 states => 1062882 policies
- Note: Full operations vector is 13 variables (110,592 states)

Sample Task

- TASK 1.1: You are working at home in the morning and plan to go directly to a meeting when you go into work. Kim said she would send you a message telling you where and when the meeting is. Find out the **Meeting Time** and the **Meeting Place**.

User Satisfaction

- In this conversation, did Elvis understand what you said? (ASR **Performance**)
- Was Elvis easy to understand in this conversation? (**TTS Performance**)
- In this conversation, was it easy to find the message you wanted? (**Task Ease**)
- Was the pace of interaction with Elvis appropriate in this conversation? (**Interaction Pace**)
- In this conversation, did you know what you could say at each point of the dialog? (**User Expertise**)
- How often was Elvis sluggish and slow to reply to you in this conversation? (**System Response**)
- Did Elvis work the way you expected him to in this conversation? (**Expected Behavior**)
- In this conversation, how did Elvis's voice interface compare to the touch-tone interface to voice mail? (**Comparable Interface**)
- From your current experience with using Elvis to get your email, do you think you'd use Elvis regularly to access your mail when you are away from your desk? (**Future Use**)

Estimating a Reward Function

- Use multivariate linear regression to model User Satisfaction as a function of Task Success and Costs
- Performance = .27* COMP + .54 * MRS - .09* Bargein% + .15 * Rejection%
  - COMP: User Perception of Task Success
  - MRS: Mean Recognition Score
  - Bargein%: BargeIns /Turns
  - Rejection%: Rejections/Turns

Value Iteration

$$U_{n+1}(a, S_i) = R(S_i) + \sum_j M_{ij}^{a} \max_{a'} U_n(a', S_j)$$

- Value Iteration: updates Un with Un+1 until difference less than a threshhold
- Threshold = 5% of performance range

Learned Policy

- Initiative: **System**
- Summary: **Summarize-System-Choice** except when TaskProg = 0.
- Read: **Read-First**

Testing the Learned Policy

- 6 users
- Same 3 tasks
- Task Completion: increased to .94 from .85
- User Satisfaction: increased to 31.7 from 27.5

Conclusions of ELVIS experiments

- First test of RL approach with human users with speech input
- Dialogue manager as an MDP provides a state-based model of user population, effects of actions
- Compared strategies for initiative and information presentation
- Measurable and significant improvement in user satisfaction

Applying Q-learning to Dialogue

$$U_{n+1}(a, S_i) = R(S_i) + \sum_j M_{ij}^a \max_{a'} U_n(a', S_j)$$

- Overall Utility U
- States Si, Sj
- Reward R
- M^a_{ij} , probability of going from state i to state j on doing action a (estimated from experimental data)
- U(final state): task completion, user satisfaction

Maximizing Expected Utility

- **Maximum Expected Utility Principle**: An optimal action is one that maximizes the expected utility of outcome states
- **Reinforcement Learning**: use rewards received at end of dialogue to learn which actions lead to highest rewards

$$U(S_i) = F(U(S_j)), S_i \prec S_j$$

Reinforcement learning

- Useful for discovering optimal strategies when rewards are clear
- Choosing reward function is important
- Should make sense, be easy to measure, and correlate with some intuition about improvement
- Data collection for reinforcement learning always an issue