Dialogue management: Parametric approaches to policy optimisation

Pei-Hao Su and Milica Gašić

Dialogue Systems Group, Cambridge University Engineering Department

February 11, 2016

Dialogue optimisation as a reinforcement learning task

Dialogue management as a continuous space Markov decision process

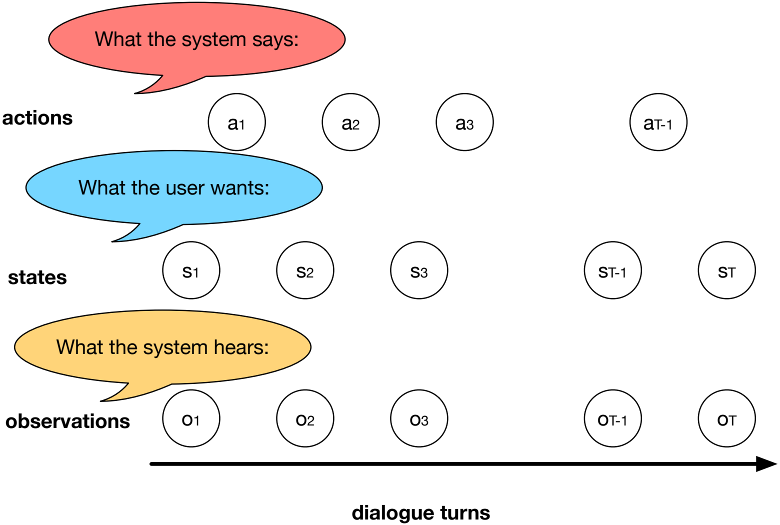
Summary space

Simulated user

RL algorithms for dialogue management

Natural Actor Critic

Elements of dialogue management



Dialogue as a control problem

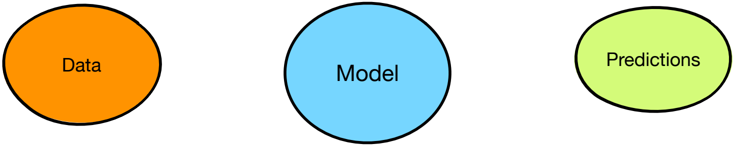
Input the distribution over possible states – belief state, the output of the belief tracker

Control actions that the system takes – what the system says to the user

Feedback signal the estimate of dialogue quality

Aim automatically optimise system actions – dialogue policy

Dialogue as a partially observable Markov decision process



- Noisy observations
- Reward – a measure of dialogue quality

- Partially observable Markov decision process

- Optimal system actions in noisy environment

Theory: Partially observable Markov decision process

s_t dialogue states

o_t noisy observations

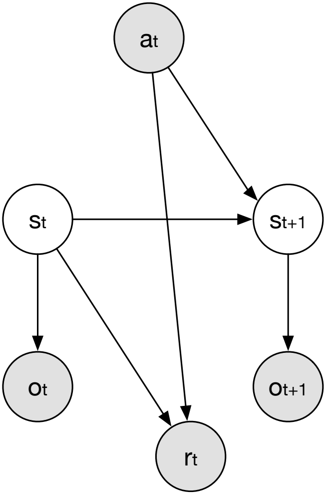
a_t system actions

r_t rewards

$p(s_{t+1}|s_t, a_t)$ transition probability

$p(o_{t+1}|s_{t+1})$ observation probability

$b(s_t)$ distribution over possible states



Decision making in POMDPs

Policy $\pi : \mathcal{B} \rightarrow \mathcal{A}$

Return $R_t = \sum_{k=0}^{T-l} \gamma^k r_{t+k}$

Value function How good is it for the system to be in a particular belief state?

$$V^\pi(s) = E_\pi \left\{ \sum_{k=0}^{T-t} \gamma^k r_{t+k} | s_t = s \right\}$$
$$= r(s, a) + \gamma \sum_{s'} p(s'|s, a) \sum_{o'} p(o'|s') V^\pi(s')$$
$$V^\pi(\mathbf{b}) = \sum_s V^\pi(s)\mathbf{b}(s)$$

Optimising POMDP policy

- Finding value function associated with optimal policy, i.e. the one that generates maximal return
- Tractable only for very simple cases [Kaelbling et al., 1998]
- Alternative view: discrete space POMDPs can be viewed as a continuous space MDP with states as belief states $b_t = b(s_t)$

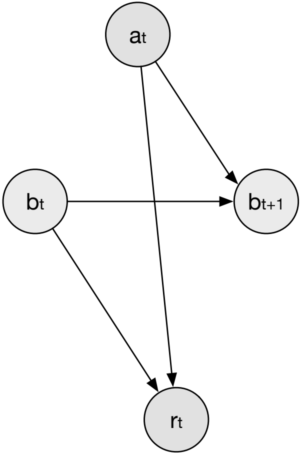
Theory: Markov decision process

b_t belief states from tracker

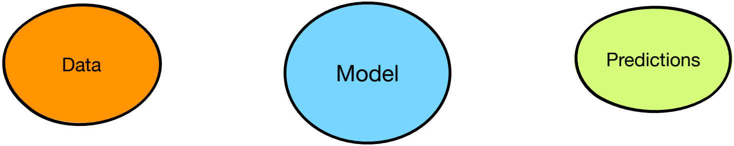
a_t system actions

r_t rewards

$p(b_{t+1}|b_t, a_t)$ transition probability



Dialogue management as a continuous space Markov decision process



- belief states (from belief tracker)
- Reward – a measure of dialogue quality

- Markov decision process and reinforcement learning

- Optimal system actions

Problems

Size of the optimisation problem

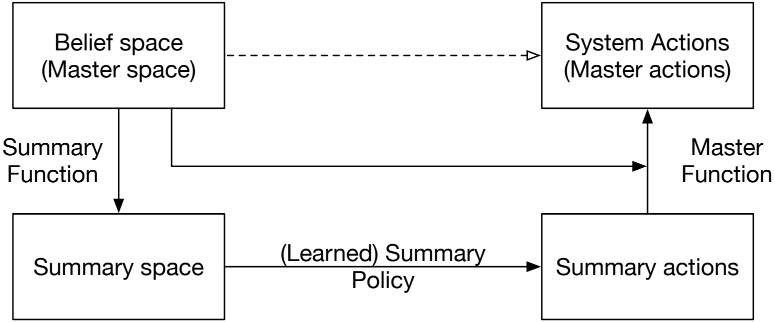
- Belief state is large and continuous
- Set of system actions also large

Knowledge of the environment, in this case the user

- We do not have transition probabilities
- Where do rewards come from?

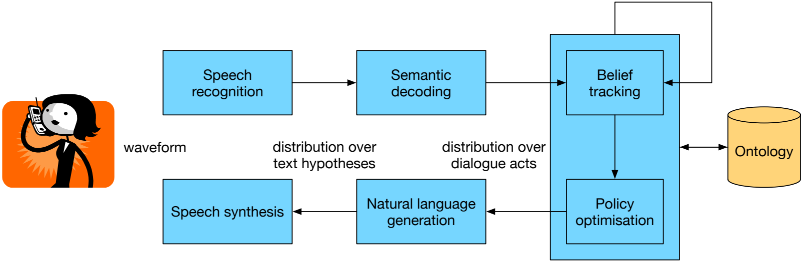
Problem: large belief state and action space

Solution: perform optimisation in a reduced space – summary space built according to the heuristics



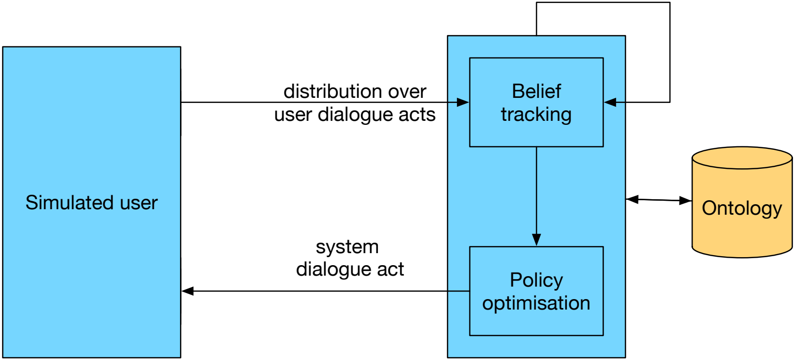
Problem: Where do the transition probability and the reward come from?

Solution: learn from real users.

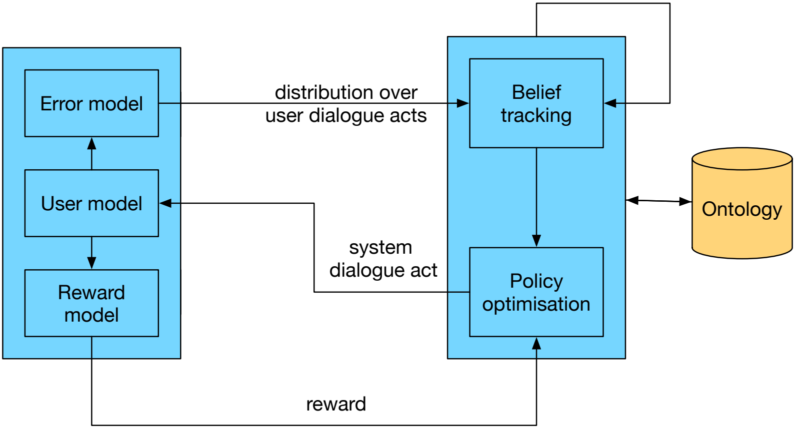


Problem: Where do the transition probability and the reward come from?

Solution: learn from a simulated user.



Elements of the simulated user



Theory: Reinforcement learning

Policy deterministic $\pi : \mathcal{B} \to \mathcal{A}$ or stochastic $\pi : \mathcal{B} \times \mathcal{A} \to [0, 1]$

Return $R_t = \sum_{k=0}^{T-l} \gamma^k r_{t+k}$

Value function How good is it for the system to be in a particular belief state?

$$V^\pi(\mathbf{b}) = E_\pi \left\{ \sum_{k=0}^{T-t} \gamma^k r_{t+k} | b_t = \mathbf{b} \right\}$$

Q-function What is the value of taking action a in belief state $\mathbf{b}$ under a policy π?

$$Q^\pi(\mathbf{b}, a) = E_\pi \left\{ \sum_{k=0}^{T-t} \gamma^k r_{t+k} | b_t = \mathbf{b}, a_t = a \right\}$$

Theory: Reinforcement learning

Occupancy frequency

$$d^{\pi}(\mathbf{b}) = \sum_{t} \gamma^{t} Pr(b_t = \mathbf{b} | \mathbf{b}_0, \pi)$$

Advantage function

$$A^{\pi}(\mathbf{b}, a) = Q^{\pi}(\mathbf{b}, a) - V^{\pi}(\mathbf{b})$$

Theory: Reinforcement learning [Sutton and Barto, 1998]

For discrete state spaces standard RL approaches can be used to estimate optimal Value function, Q-function or policy π

Dynamic programming
: is model-based learning and update of the estimates are based on the previous estimates

Monte-Carlo methods
: is model-free learning and update of estimates based is based on raw experience

Temporal-difference methods
: is model-free learning and update of the estimates are based on the previous estimates

Reinforcement learning for dialogue management

Options

1. Discretise the belief state/summary space into a grid and apply standard RL algorithms to estimate Value function, Q-function or policy π (for example Monte-Carlo Control in practical)
2. Apply parametric function approximation to Value function, Q-function or policy π and find optimal parameters using gradient methods (this lecture)
3. Apply non-parametric function approximation to Value function, Q-function or policy π (next lecture)

Linear function approximation

Define summary space as features of belief space (ϕ or ϕ_a) and parameterise either:

- Value function
$$V(\mathbf{b}, \boldsymbol{\theta}) \approx \boldsymbol{\theta}^\mathsf{T} \phi(\mathbf{b})$$
- Q-function
$$Q(\mathbf{b}, a, \boldsymbol{\theta}) \approx \boldsymbol{\theta}^\mathsf{T} \phi_a(\mathbf{b})$$
- policy
$$\pi(a|\mathbf{b}, \boldsymbol{\theta}) = \frac{e^{\boldsymbol{\theta}^\mathsf{T} \phi_a(\mathbf{b})}}{\sum_{a'} e^{\boldsymbol{\theta}^\mathsf{T} \phi_{a'}(\mathbf{b})}}$$

Policy gradient

- Find policy parameters that maximise return $J(\boldsymbol{\theta}) = E_{\pi(\boldsymbol{\theta})}\left\{R_0\right\}$
- Update policy parameters in the direction of gradient $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \nabla J(\boldsymbol{\theta})$ – *vanilla* gradient given by policy gradient theorem [Sutton et al., 2000]

$$\nabla J(\boldsymbol{\theta}) = \int_{\mathcal{B}} d^{\pi}(\mathbf{b}) \sum_{a} Q^{\pi}(\mathbf{b}, a)\pi(\mathbf{b}, a)\nabla log \pi(\mathbf{b}, a) d\mathbf{b} \quad (1)$$

$$= E_{\pi(\boldsymbol{\theta})}\left\{\nabla_{\boldsymbol{\theta}} \log \pi(b, a) Q^{\pi(\boldsymbol{\theta})}(b, a)\right\} \quad (2)$$

$$= E_{\pi(\boldsymbol{\theta})}\left\{\nabla_{\boldsymbol{\theta}} \log \pi(b, a) A^{\pi(\boldsymbol{\theta})}(b, a)\right\} \quad (3)$$

- This is not always stable – (large) changes in the parameters can result in unexpected policy moves.
- Convergence can be very slow.

Natural Actor Critic [Peters and Schaal, 2008, Thomson, 2009]

Actor-critic methods are Temporal-difference methods that estimate

actor
: policy that takes actions parametrised with $\boldsymbol{\theta}$

critic
: Advantage function that criticises/evaluates actor actions parameterised with $\boldsymbol{\omega}$

In Natural Actor Critic

- Critic reduces the variance – the learning is more stable
- A modified form of gradient – *natural gradient* is used to find the optimal parameters to speed up the convergence.

Natural Policy Gradient

Compatible function approximation

- Advantage function is parametrised with parameters $\boldsymbol{\omega}$ such that the direction of change is the same as for the policy parameters $\boldsymbol{\theta}$

$$\nabla_{\boldsymbol{\omega}} A_{\boldsymbol{\omega}}(\mathbf{b}, a) = \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{b}, a)$$

- Then by replacing

$$A_{\boldsymbol{\omega}}(\mathbf{b}, a) = \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{b}, a)^{\mathsf{T}} \boldsymbol{\omega}$$

  in Eq 3
- It can be shown

$$\boldsymbol{\omega} = G_{\boldsymbol{\theta}}^{-1} \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$

  where $G_{\boldsymbol{\theta}}$ is the Fisher information matrix

$$G_{\boldsymbol{\theta}} = E_{\pi(\boldsymbol{\theta})}(\nabla \log \pi_{\boldsymbol{\theta}}(\mathbf{b}, a) \nabla \log \pi_{\boldsymbol{\theta}}(\mathbf{b}, a)^{\mathsf{T}})$$

Episodic Natural Actor Critic

Algorithm 1 Episodic Natural Actor Critic

for each batch of dialogues **do**
  for each dialogue n **do**
    Execute the dialogue according to the current policy $\pi(\boldsymbol{\theta})$
    Obtain sequence of belief states, actions and corresponding rewards
  end for
  Critic evaluation Choose $\boldsymbol{\omega}, J$ to minimise $\sum_n (A_{\boldsymbol{\omega}} + J - R_n)^2$
  Actor update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \boldsymbol{\omega}$
end for

Summary features

- For each concept the probability of two most likely values mapped into a grid
- Number of matching entities in the database (assuming most likely concepts)
- A parameter is associated with each summary action, concept and concept level feature
- Parameters can be tied to reduce computational complexity and over-fitting

Summary

- Dialogue policy optimisation can be viewed as a reinforcement learning task
- POMDP can be viewed as a continuous space MDP
- Belief state space can be summarised to reduce computational complexity
- Natural Actor Critic is a temporal-difference algorithm which estimates both the policy (actor) and the Q-function (critic).
- Both policy and Q-function are parametrised and natural gradient is used to find the direction of the steepest descent

Natural gradient [Amari, 1998]

- Distance in Riemann space: $|d\boldsymbol{\theta}|^2 = d\boldsymbol{\theta}^\mathsf{T} G_{\boldsymbol{\theta}} d\boldsymbol{\theta}$, where $G_{\boldsymbol{\theta}}$ is a metric tensor
- Direction of steepest descent in Riemann space for some loss function $L(\boldsymbol{\theta})$ is $G_{\boldsymbol{\theta}}^{-1} \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$
- If $\boldsymbol{\theta}$ is used to optimise the estimate of a probability distribution $p(x|\boldsymbol{\theta})$ then the optimal metric tensor is Fisher information matrix as this give distances invariant to scaling of the parameters.

$$G_{\boldsymbol{\theta}} = E(\nabla \log p(x|\boldsymbol{\theta}) \nabla \log p(x|\boldsymbol{\theta})^\mathsf{T})$$

- It can be shown that $KL(p(x|\boldsymbol{\theta})||p(x|\boldsymbol{\theta} + d\boldsymbol{\theta})) \approx d\boldsymbol{\theta}^\mathsf{T} G_{\boldsymbol{\theta}} d\boldsymbol{\theta}$

References I

Amari, S.-I. (1998).
Natural gradient works efficiently in learning.
Neural Comput., 10(2):251–276.

Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998).
Planning and acting in partially observable stochastic domains.
Artif. Intell., 101(1-2):99–134.

Peters, J. and Schaal, S. (2008).
Natural actor-critic.
Neurocomputing, 71(7):1180–1190.

Sutton, R. S. and Barto, A. G. (1998).
Introduction to Reinforcement Learning.
MIT Press, Cambridge, MA, USA, 1st edition.

References II

Sutton, R. S., Mcallester, D., Singh, S., and Mansour, Y. (2000).
Policy gradient methods for reinforcement learning with function approximation.
In *In Advances in Neural Information Processing Systems 12*, pages 1057–1063. MIT Press.

Thomson, B. (2009).
Statistical methods for spoken dialogue management.
PhD thesis, University of Cambridge.