Deep Reinforcement Learning for Dialogue Management

Paweł Budzianowski
Dialogue Systems Group
Cambridge University

Conversational agents

1. Non-task oriented dialogues - sequence-to-sequence

   Vinyals & Le, 2015

   Serban et al., 2015

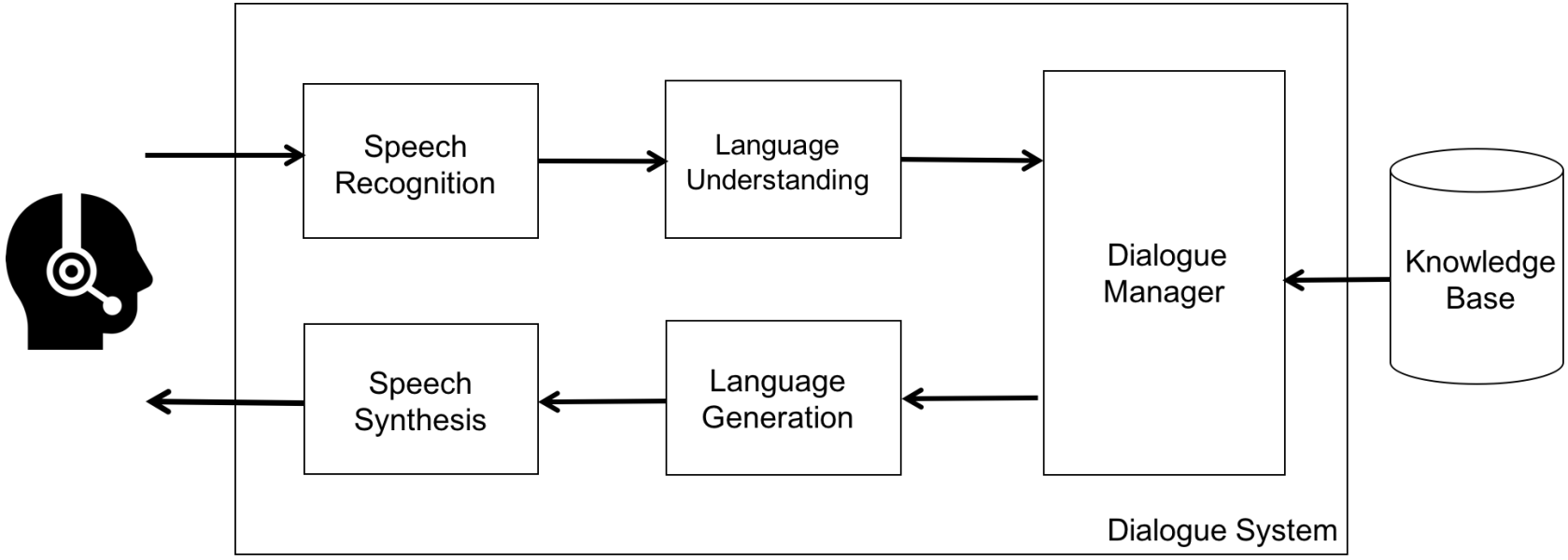
   Li et al, 2016, 2017

Conversational agents

1. Non-task oriented dialogues - sequence-to-sequence
   Vinyals & Le, 2015
   Serban et al., 2015
   Li et al, 2016, 2017
2. Task oriented dialogues - modular approaches
   Raux et al., 2005
   Young et al 2006, 2013

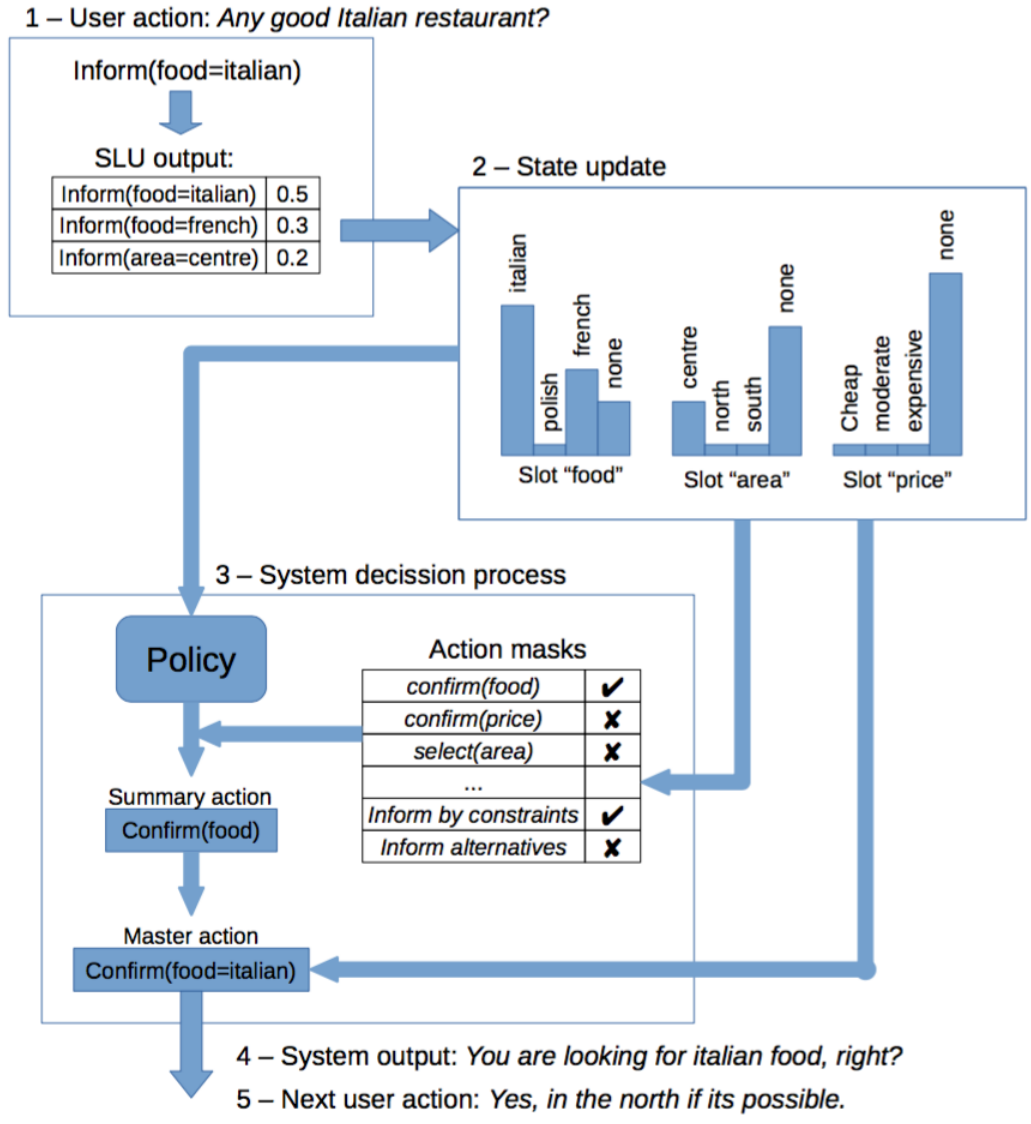
Spoken Dialogue Systems

Spoken Dialogue Systems



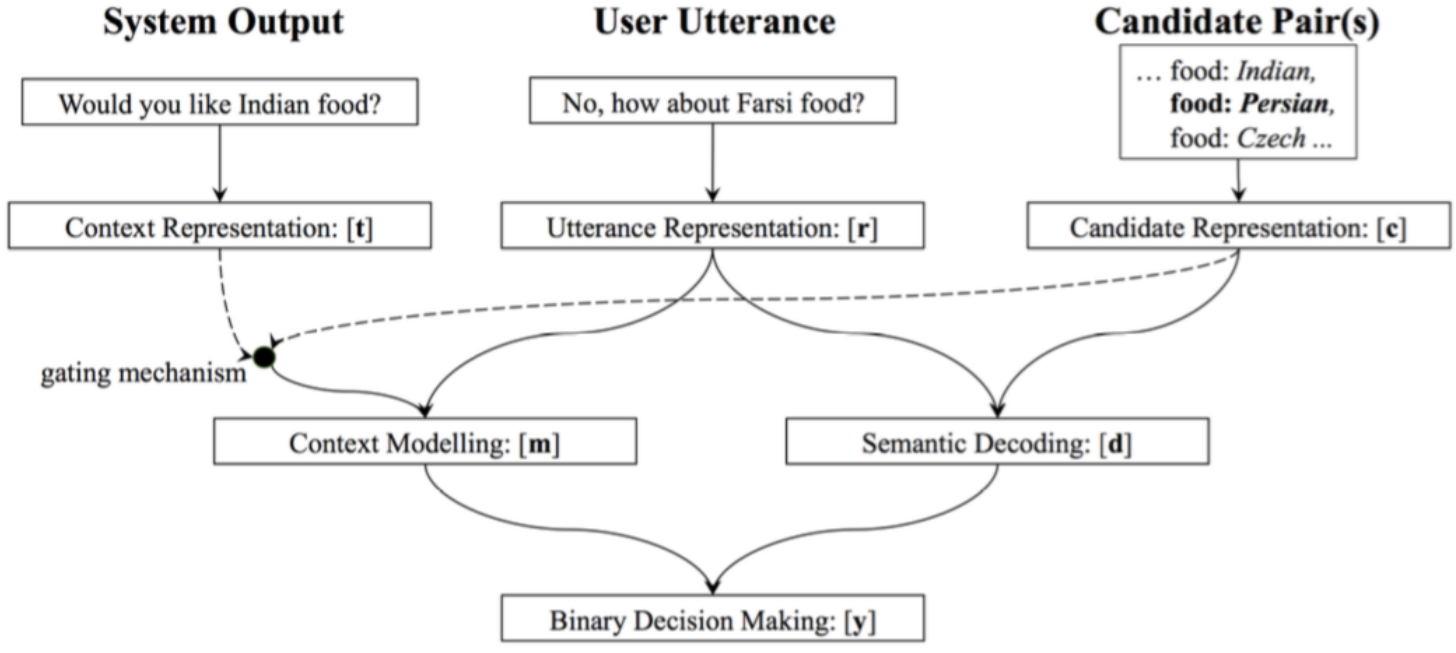
Dialog flow

Dialog flow


1 – User action: *Any good Italian restaurant?*
Inform(food=italian)
SLU output:
Inform(food=italian) 0.5
Inform(food=french) 0.3
Inform(area=centre) 0.2
2 – State update
italian
polish
french
none
Slot "food"
centre
north
south
none
Slot "area"
Cheap
moderate
expensive
none
Slot "price"
3 – System decission process
Policy
Action masks
confirm(food) ✔
confirm(price) ✘
select(area) ✘
...
Inform by constraints ✔
Inform alternatives ✘
Summary action
Confirm(food)
Master action
Confirm(food=italian)
4 – System output: *You are looking for italian food, right?*
5 – Next user action: *Yes, in the north if its possible.*

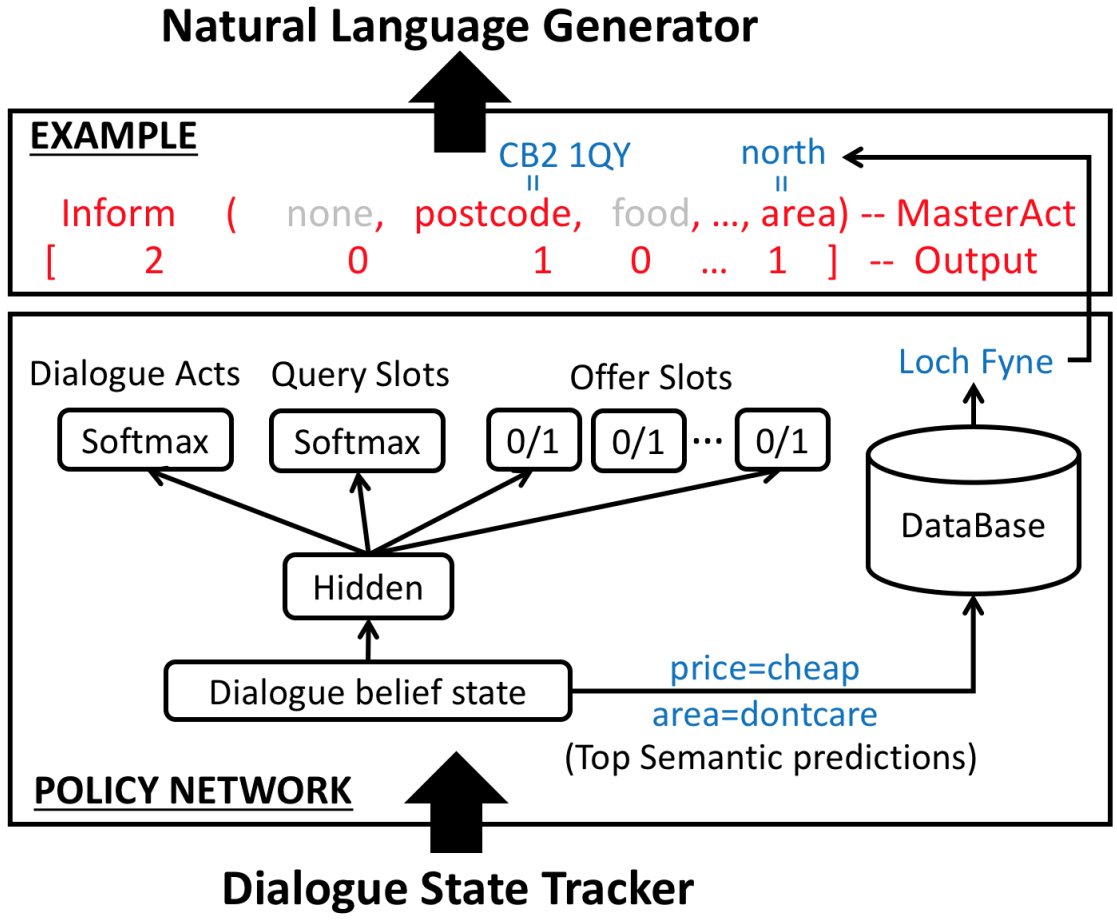
Natural Language Understanding

Natural Language Understanding



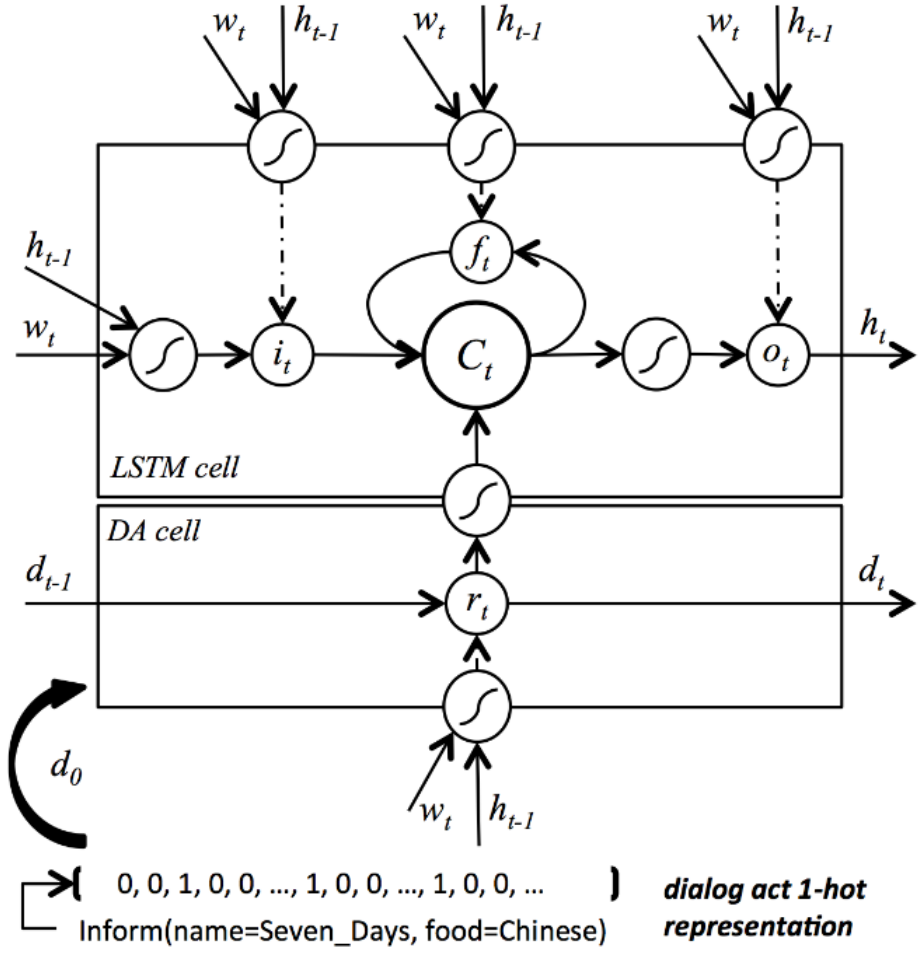
Dialogue manager

Dialogue manager



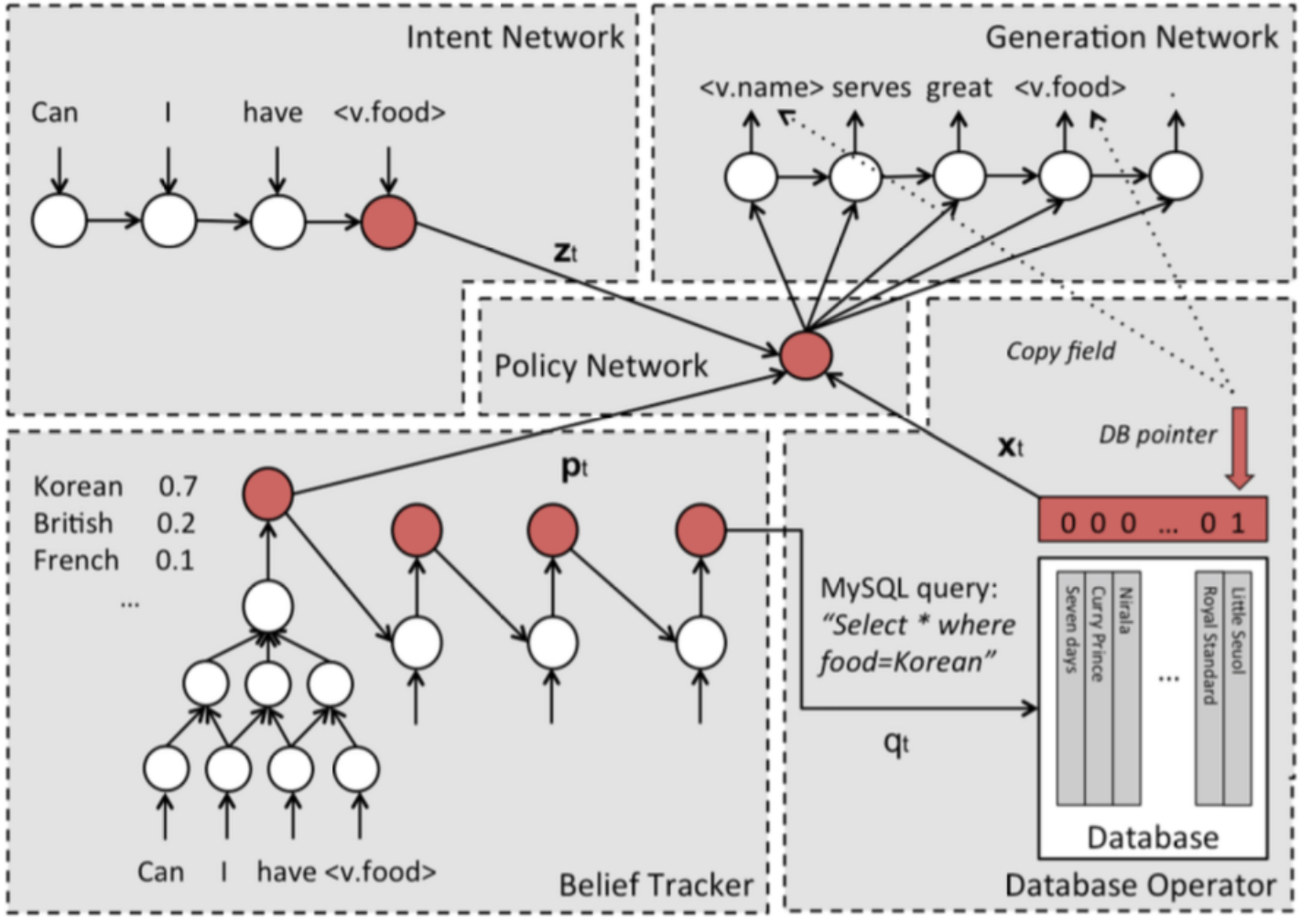
Language generation

Language generation



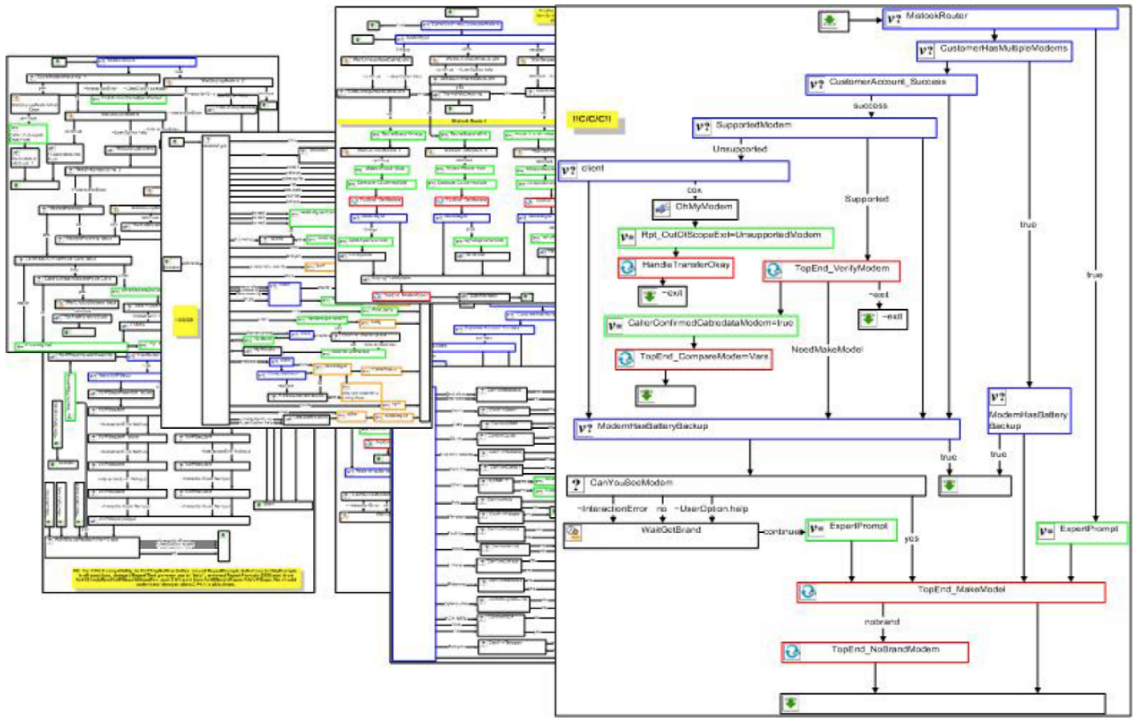
Combining them all?

Combining them all?



Policy/Dialogue Management

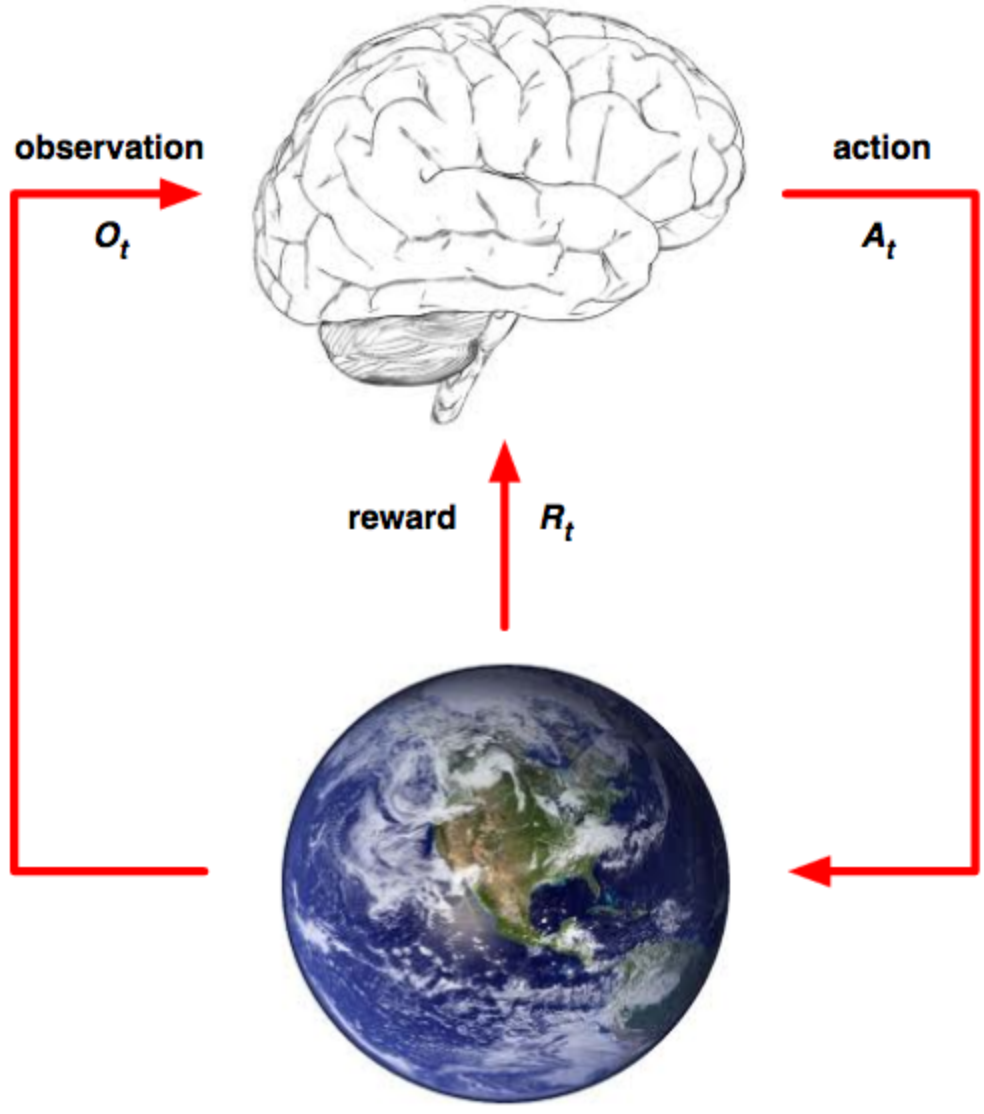
Policy/Dialogue Management



Answer?

Reinforcement learning

Reinforcement learning



Value-based RL

We approximate the expected discounted sum of the future rewards received over the course of a dialogue given an action a in a belief state b:

Value-based RL

We approximate the expected discounted sum of the future rewards received over the course of a dialogue given an action a in a belief state b:

$$Q_t(b, a) = \mathbb{E}_\pi\{r_t + \gamma r_{t+1} + \ldots \ |b_t = b, a_t = a\}.$$

Policy-based RL

However, we can move directly to the main objective we care about which is the policy. The objective is to find a policy that maximizes the expected discounted return:

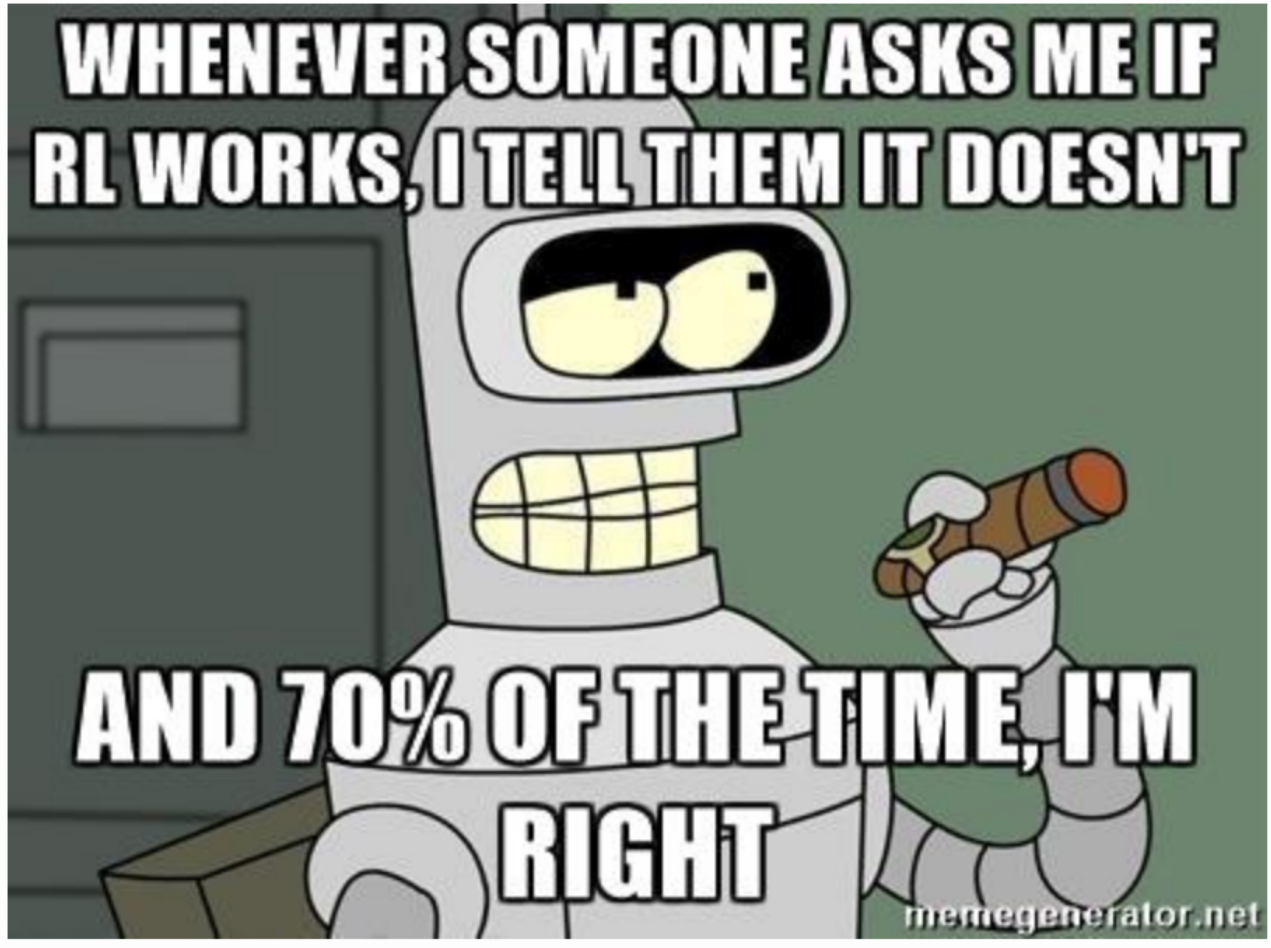
Policy-based RL

However, we can move directly to the main objective we care about which is the policy. The objective is to find a policy that maximizes the expected discounted return:

$$J(\theta) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{n} r_t\right].$$

Does RL work?

Does RL work?


WHENEVER SOMEONE ASKS ME IF RL WORKS, I TELL THEM IT DOESN'T
AND 70% OF THE TIME, I'M RIGHT
memegenerator.net

Problems of Deep RL in dialogue policy optimisation

1. Cold start problem,

Problems of Deep RL in dialogue policy optimisation

1. Cold start problem,
2. Sample efficiency in large action space,

Problems of Deep RL in dialogue policy optimisation

1. Cold start problem,
2. Sample efficiency in large action space,
3. No uncertainty estimates.

Problems of Deep RL in dialogue policy optimisation

3. No uncertainty estimates.

No uncertainty estimates...

No uncertainty estimates...

1. Osband et al. 2016, (Stanford, Deepmind)

No uncertainty estimates...

1. Osband et al. 2016, (Stanford, Deepmind)
2. Lipton et al., 2017, (CMU, Microsoft)

No uncertainty estimates...

1. Osband et al. 2016, (Stanford, Deepmind)
2. Lipton et al., 2017 (CMU, Microsoft)
3. Fortunato et al., 2017 (Berkley, Deepmind)

No uncertainty estimates...

1. Osband et al. 2016, (Stanford, Deepmind)
2. Lipton et al., 2017 (CMU, Microsoft)
3. Fortunato et al., 2017 (Berkley, Deepmind)
4. Azizzadenesheli et al., 2018 (Stanford)

Improving Exploration through Bayesian Deep RL

with Chris Tegho (Calipsa)

Exploration vs exploitation dilemma

Online decision-making involves a fundamental choice between:

Exploration vs exploitation dilemma

Online decision-making involves a fundamental choice between

1. **exploitation** - make the best decision given current information

Exploration vs exploitation dilemma

Online decision-making involves a fundamental choice between

1. **exploitation** - make the best decision given current information

versus

2. **exploration** - gather more information.

Prior vs posterior

Bayes rule:

$$P(\mathbf{w}|\mathcal{D})$$

Prior vs posterior

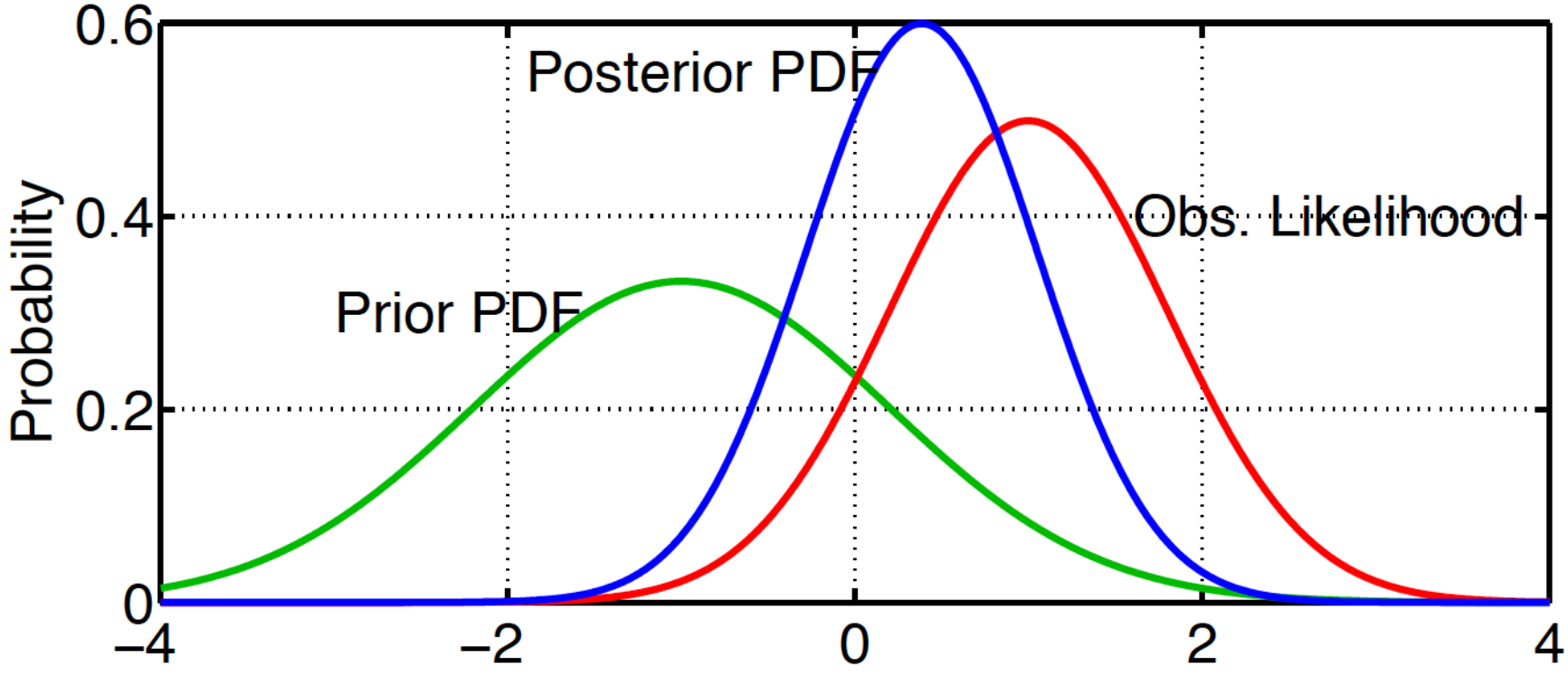
Bayes rule:

$$P(\mathbf{w}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathbf{w})P(\mathbf{w})}{P(\mathcal{D})}$$

Prior vs posterior

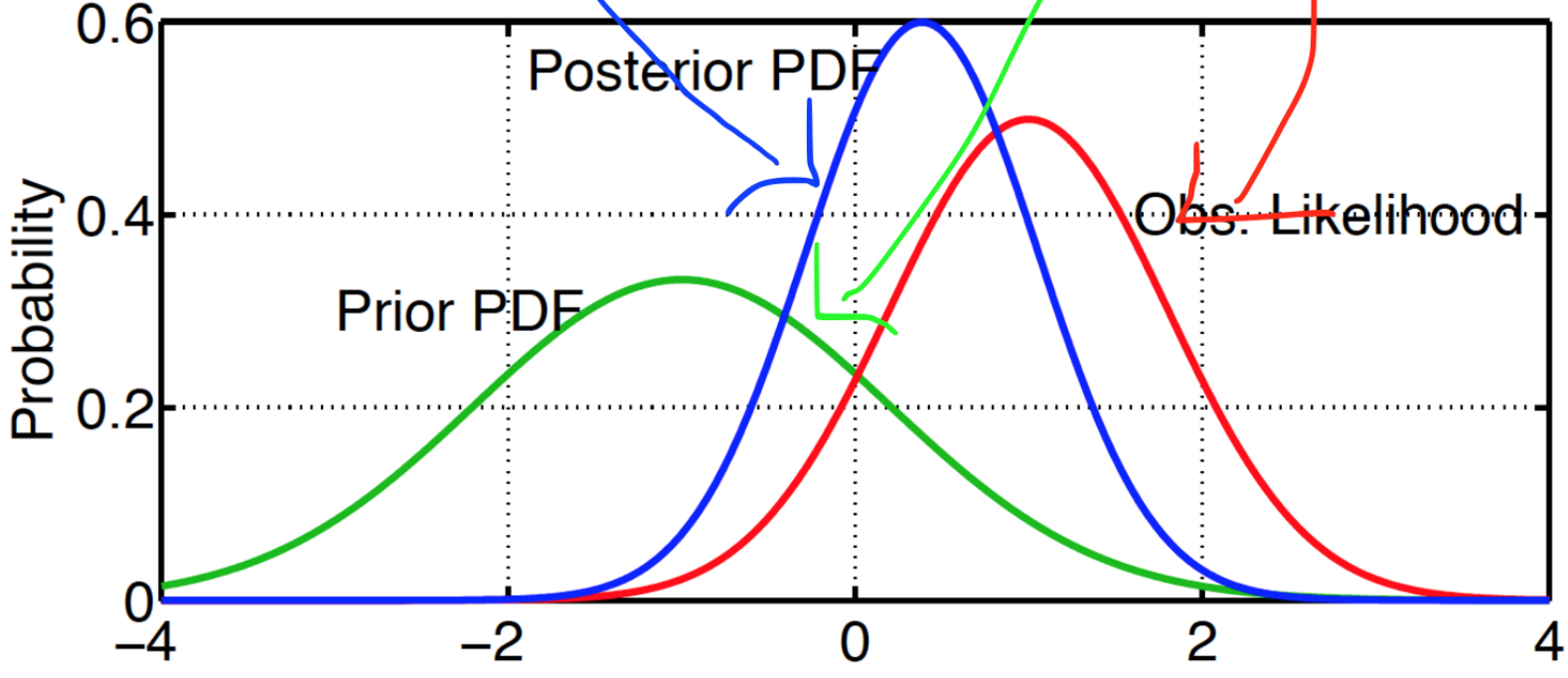
Bayes rule:

$$P(\mathbf{w}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathbf{w})P(\mathbf{w})}{P(\mathcal{D})}$$



Prior vs posterior

$$P(\mathbf{w}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathbf{w})P(\mathbf{w})}{P(\mathcal{D})}$$



Thompson Sampling

Assume we have some posterior over parameters of our model $P(w|\mathcal{D})$ given observed data $\mathcal{D}$.

Thompson Sampling

Assume we have some posterior over parameters of our model $P(w|\mathcal{D})$ given observed data $\mathcal{D}$.

We can ask for the following quantity:

$$P(Q|b) = \mathbb{E}_{P(w|\mathcal{D})}[P(Q|b,w)].$$

Q-learning

$$Q^{new}(b_t, a_t) \leftarrow (1 - \alpha) Q^{old}(b_t, a_t) + \alpha(r_t + \gamma \max_a Q(b_{t+1}, a)$$

DQN

The Deep Q-Network (DQN) algorithm models action-value function $Q(b, a)$ using a deep neural network $\hat{Q}(b, a; \theta)$, with weight vector $\theta \in \mathbb{R}^n$.

DQN

The Deep Q-Network (DQN) algorithm models action-value function $Q(b, a)$ using a deep neural network $\hat{Q}(b, a; \theta)$, with weight vector $\theta \in \mathbb{R}^n$.

DQN

We iteratively improve the prediction by minimizing the following loss:

$$L(\theta_t) = \mathbb{E}\left[(y_t - \hat{Q}(b_t, a_t; \theta_t))^2\right],$$

where the targets y_t are:

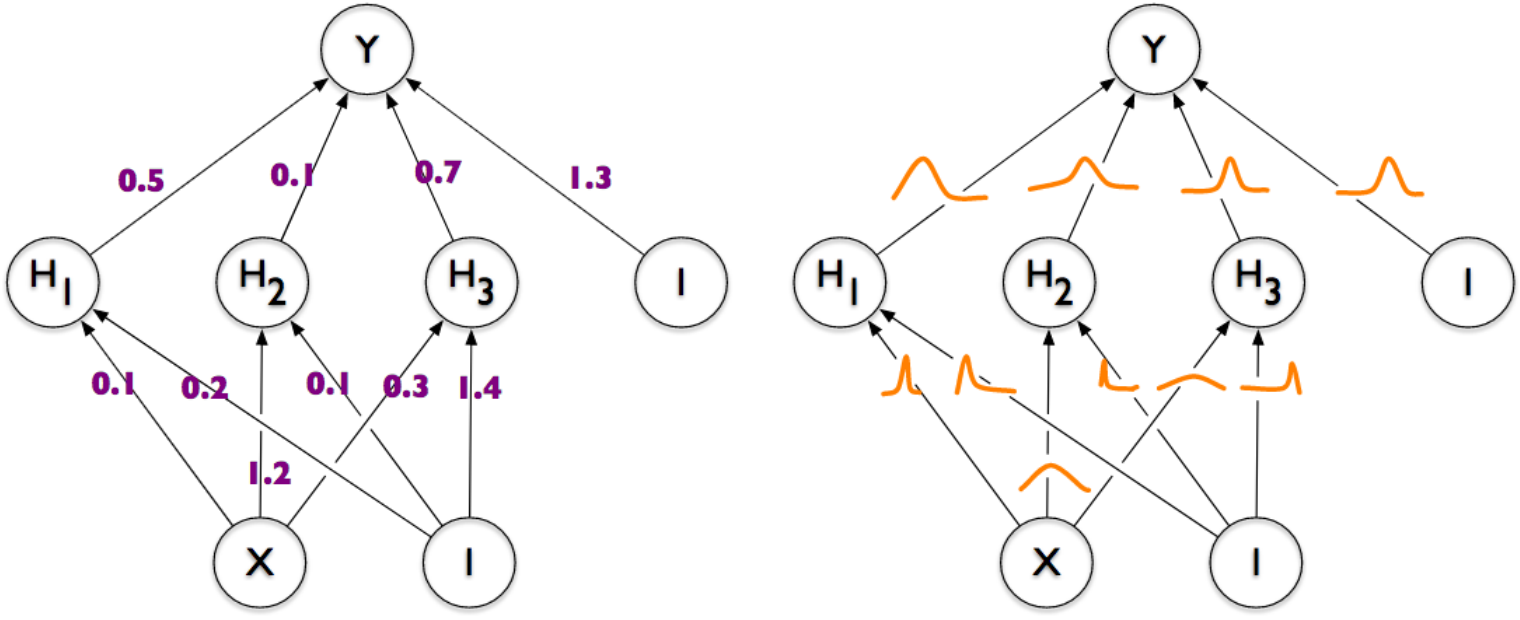
$$y_t = r_t + \gamma \max_{a'} \hat{Q}(s_{t+1}, a'; \theta_t).$$

Uncertainty estimates in neural networks

Uncertainty estimates in neural networks

Instead of having single fixed value weights w in the neural networks, all weights are represented by probability distributions over possible values given observed dialogues $\mathcal{D}$, $P(w|\mathcal{D})$.

Uncertainty estimates in neural networks



Variational inference

An intractable posterior $P(\mathbf{w}|\mathcal{D})$ is approximated by $q(\mathbf{w}|\theta)$ using variational inference approach:

$$\theta^* = \arg\min_{\theta} \mathrm{KL}[q(\mathbf{w}|\theta)||P(\mathbf{w}|\mathcal{D})]$$

Variational inference

An intractable posterior $P(\mathbf{w}|\mathcal{D})$ is approximated by $q(\mathbf{w}|\theta)$ using variational inference approach:

$$\theta^* = \arg\min_\theta \mathrm{KL}[q(\mathbf{w}|\theta)||P(\mathbf{w}|\mathcal{D})]$$

$$= \arg\min_\theta \mathbb{E}_{q(\mathbf{w}|\theta)} \left[\frac{\log P(q(\mathbf{w}|\theta))}{\log(P(\mathcal{D}|\mathbf{w})P(\mathbf{w})/P(\mathcal{D}))} \right]$$

Variational inference

An intractable posterior $P(\mathbf{w}|\mathcal{D})$ is approximated by $q(\mathbf{w}|\theta)$ using variational inference approach:

$$\begin{aligned}\theta^* &= \arg\min_\theta \mathrm{KL}[q(\mathbf{w}|\theta)||P(\mathbf{w}|\mathcal{D}) \\ &= \arg\min_\theta \mathbb{E}_{q(\mathbf{w}|\theta)}\left[\frac{\log P(q(\mathbf{w}|\theta))}{\log(P(\mathcal{D}|\mathbf{w})P(\mathbf{w})/P(\mathcal{D}))}\right] \\ &= \arg\min_\theta \mathrm{KL}\left[q(\mathbf{w}|\theta)||P(\mathbf{w})\right] - \mathbb{E}_{q(\mathbf{w}|\theta)}[\log P(\mathcal{D}|\mathbf{w})].\end{aligned}$$

Variational inference

Usually $q(\mathbf{w}|\theta)$ is a diagonal Gaussian distribution and our network models variational posterior parameters, i.e. mean and standard deviation.

Bayes-By-Backprop

We choose $q(w|\theta)$ to be a Gaussian with diagonal covariance with a variational parameter set θ.

Bayes-By-Backprop

We choose $q(w|\theta)$ to be a Gaussian with diagonal covariance with a variational parameter set θ.

Given the mean μ_i and covariance σ_i of q for each weight, a sample from a distribution is obtained by first sampling $\epsilon_i \sim \mathcal{N}(0, \sigma_\epsilon)$, then computing:

$$w_i = \mu_i + \sigma_i \circ \epsilon_i.$$

Bayes-By-Backprop

The resulting gradient estimator of the variational objective is unbiased and has a lower variance. The exact cost can then be approximated as:

Bayes-By-Backprop

The resulting gradient estimator of the variational objective is unbiased and has a lower variance. The exact cost can then be approximated as:

$$\mathcal{F}(\mathcal{D}, \theta) \approx \sum_{i=1}^{n} \log q(w^{(i)}|\theta) - \log P(w^{(i)}) - \log p(\mathcal{D}|w^{(i)})$$

Alpha-Divergence

The α-divergence measures the similarity between two distributions and can take the form:

Alpha-Divergence

The α-divergence measures the similarity between two distributions and can take the form:

$$D_\alpha[p||q] = \frac{1}{\alpha(\alpha-1)}(1 - \int p(\theta)^\alpha q(\theta)^{1-\alpha} d\theta),$$

where $\alpha \neq 0$.

Alpha-Divergence

The approximate inference technique described in the Bayes By Backprop method corresponds to Variational Bayes (VB), which is a particular case of α-divergence, where $\alpha \rightarrow 0$.

Dropout

Dropout consists of randomly dropping units (with some probability p) from the neural network during training.

As in the previous methods, dropout can be analyzed from the variational inference perspective.

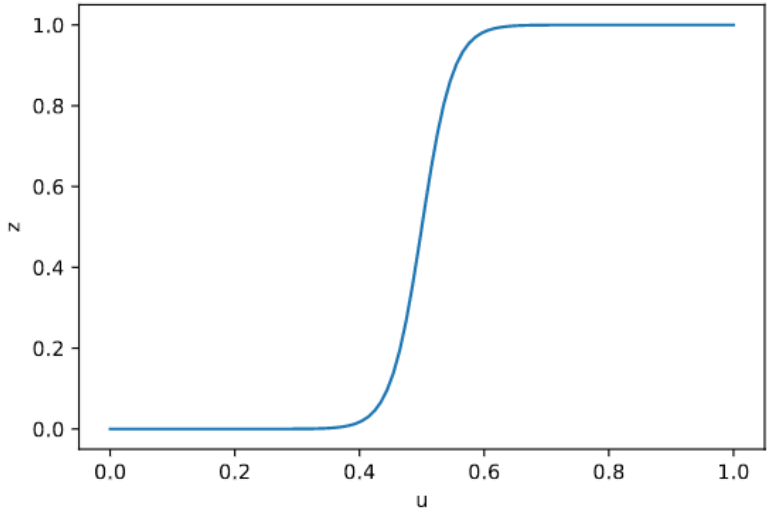
Concrete-Dropout

The concrete distribution relaxation z of the Bernoulli random variable becomes:

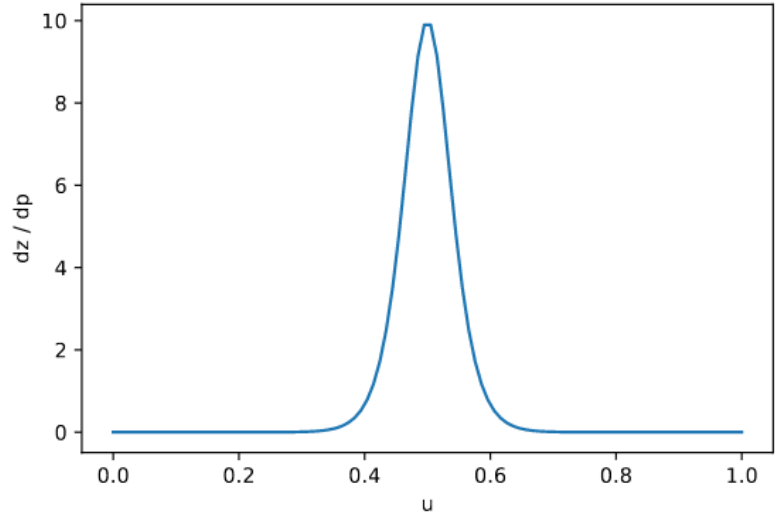
$$z = \text{sigmoid}(\frac{1}{t}(\log p - \log(1-p) + \log u - \log(1-u))$$

with some temperature t which results in values in the interval $[0, 1]$ and $u \sim \mathcal{U}(0, 1)$.

Concrete-Dropout



$$z \sim \text{Concrete}(p)$$



$$\frac{\mathrm{d}z}{\mathrm{d}p}$$

Bootstrapped method

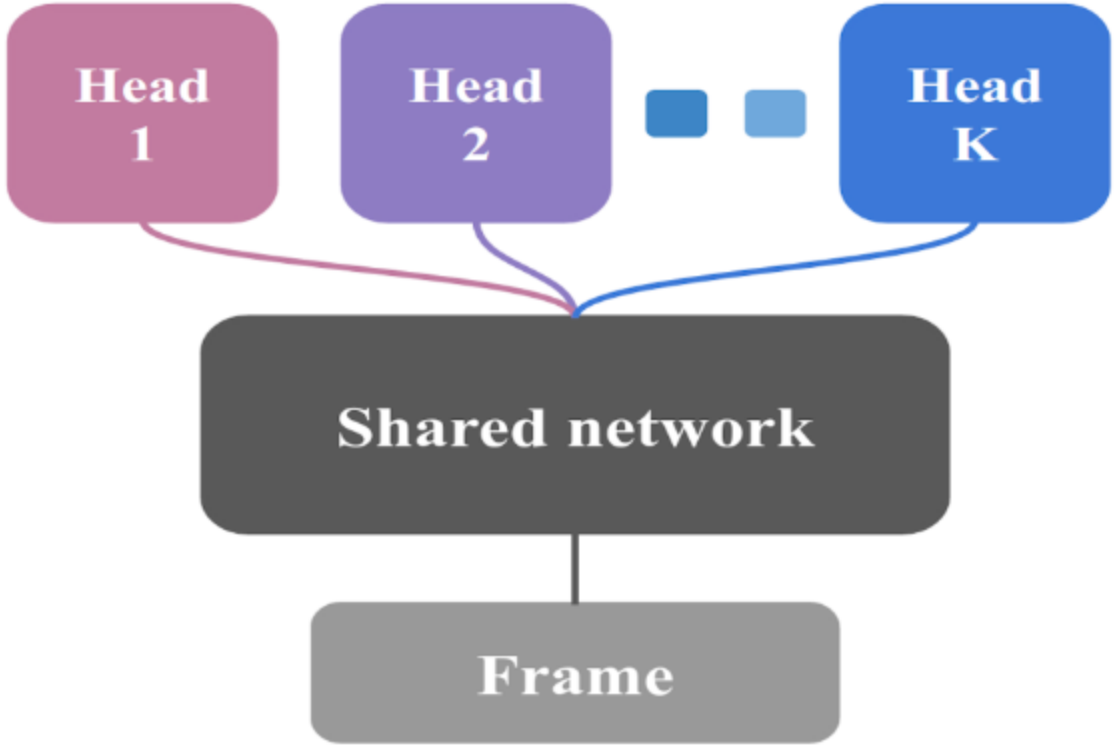
Uncertainty estimates can be obtained with random initialization of several neural networks which predict in ensemble uncertainty estimates for neural networks.

Bootstrapped method

Uncertainty estimates can be obtained with random initialization of several neural networks which predict in ensemble uncertainty estimates for neural networks.

To improve efficiency, all networks share the same architecture with a different last layer (head) computing Q-values.

Bootstrapped method

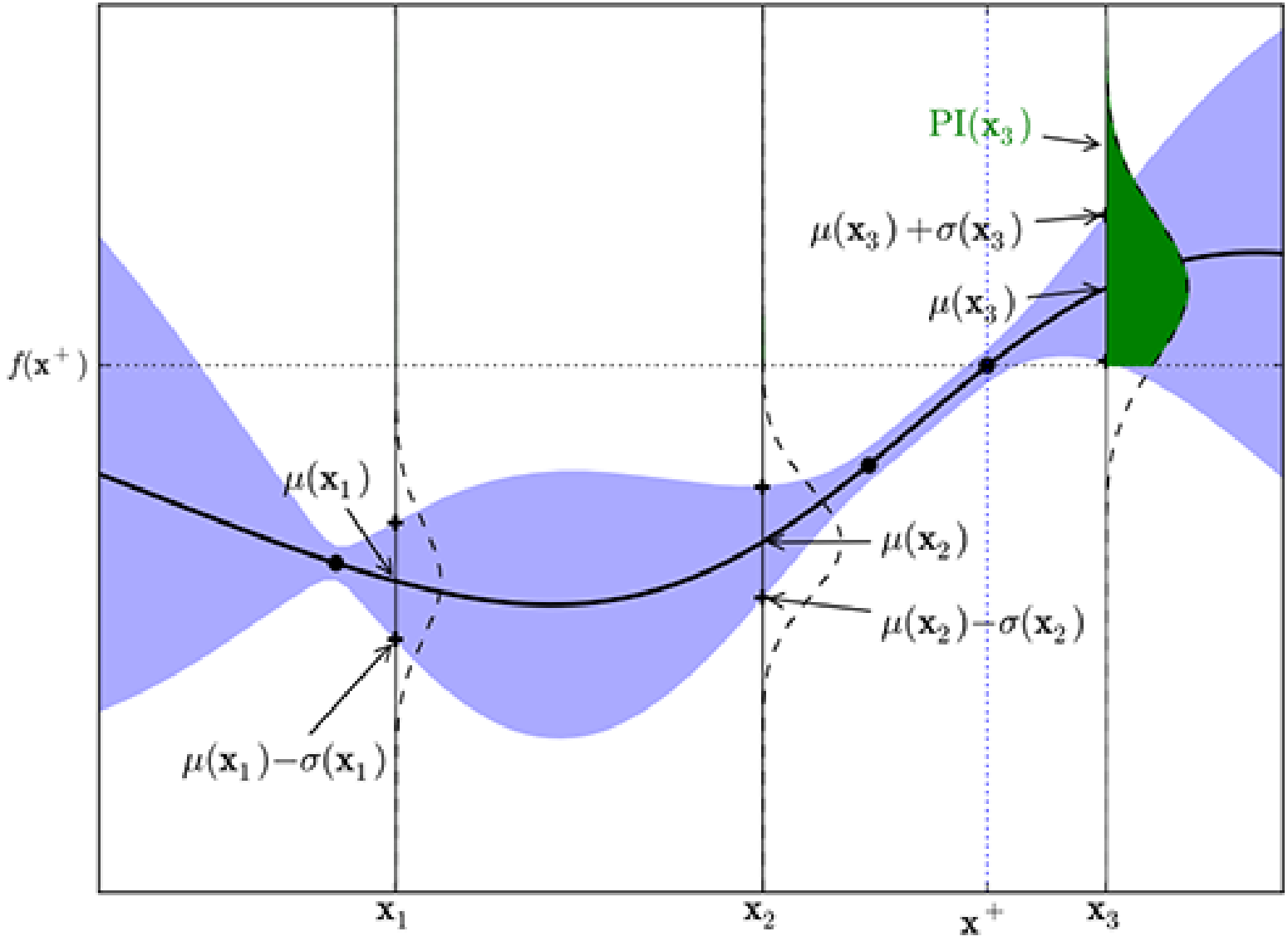


GPSARSA

Gaussian Processes SARSA was used as a fully non-parametric benchmark:

$$Q(b, a) \sim \mathcal{GP}\left(0, (k(b, a), k(b, a))\right).$$

GPSARSA


PI($\mathbf{x}_3$)
$\mu(\mathbf{x}_3)+\sigma(\mathbf{x}_3)$
$\mu(\mathbf{x}_3)$
$f(\mathbf{x}^+)$
$\mu(\mathbf{x}_1)$
$\mu(\mathbf{x}_2)$
$\mu(\mathbf{x}_2)-\sigma(\mathbf{x}_2)$
$\mu(\mathbf{x}_1)-\sigma(\mathbf{x}_1)$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}^+$
$\mathbf{x}_3$

Evaluation set-up

Cambridge restaurant domain: 100 venues, 6 slots, 3 requestable.

Evaluation set-up

Cambridge restaurant domain: 100 venues, 6 slots, 3 requestable. Belief state input of size 268 (last system act, distribution over user intent ...).

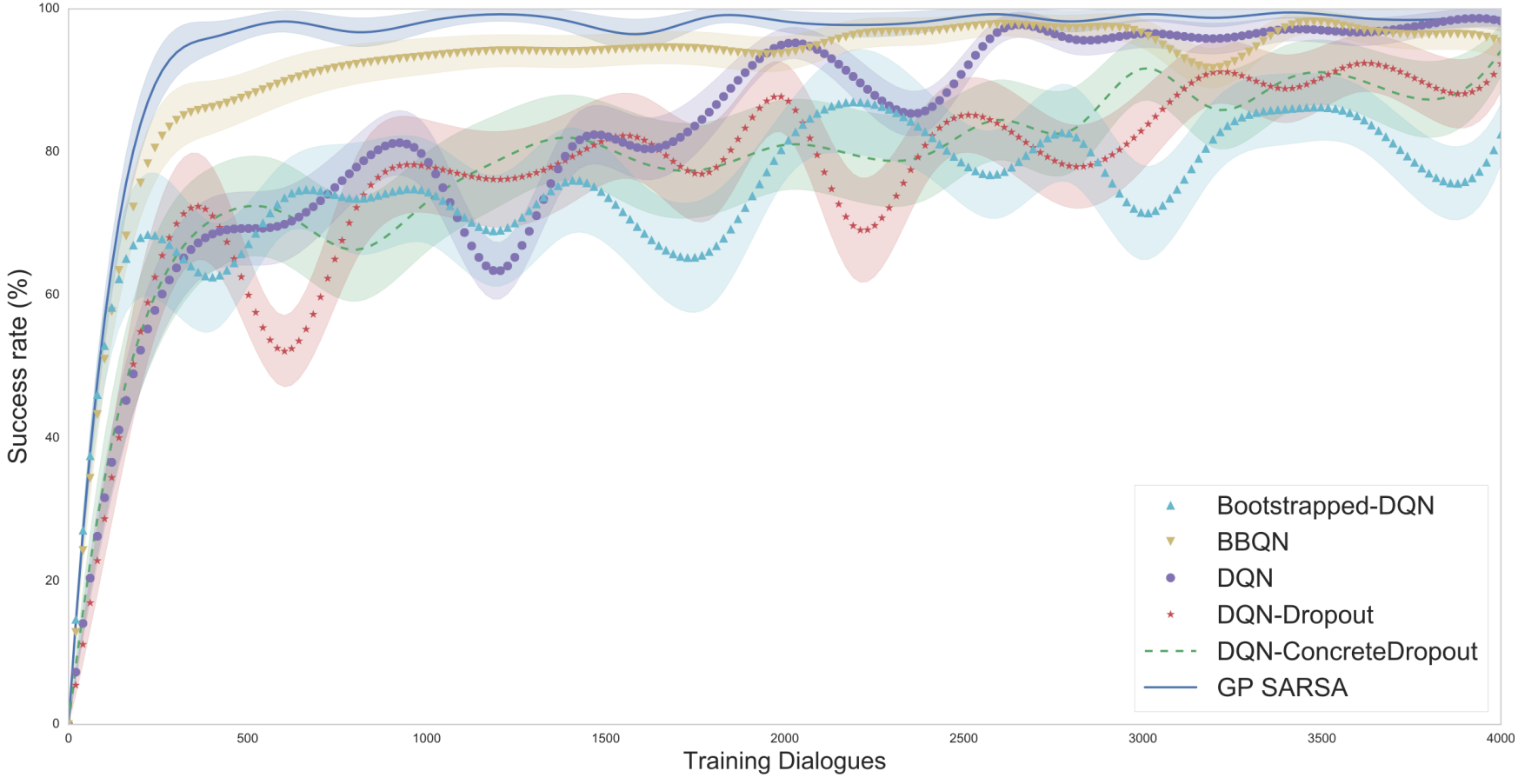
Evaluation set-up

Cambridge restaurant domain: 100 venues, 6 slots, 3 requestable.
Belief state input of size 268 (last system act, distribution over user intent ...).
System summary action space of size 14 (inform, request, confirm, ...).

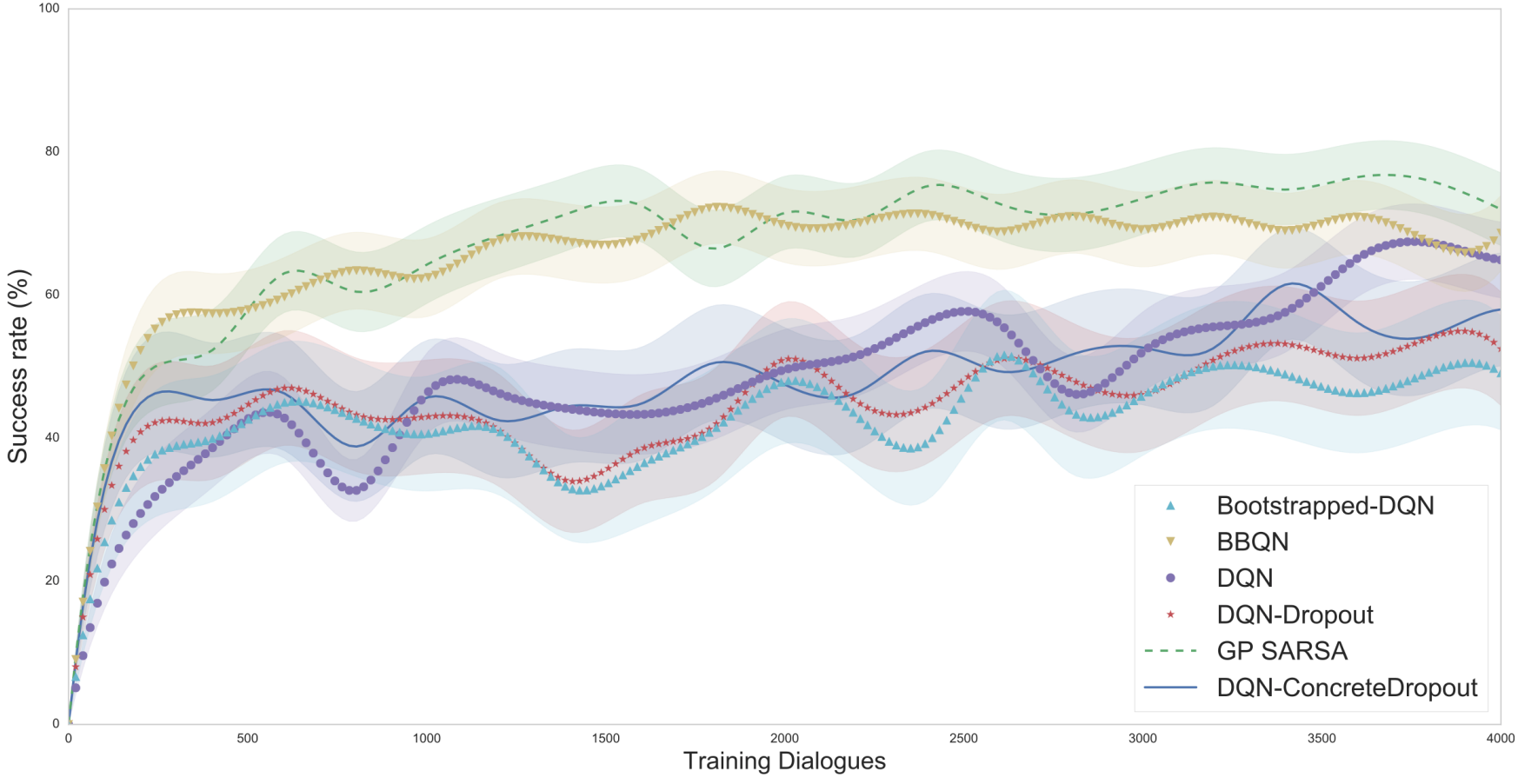
Evaluation set-up

Cambridge restaurant domain: 100 venues, 6 slots, 3 requestable.
Belief state input of size 268 (last system act, distribution over user intent ...).
System summary action space of size 14 (inform, request, confirm, ...).
User simulator operating on semantic level.

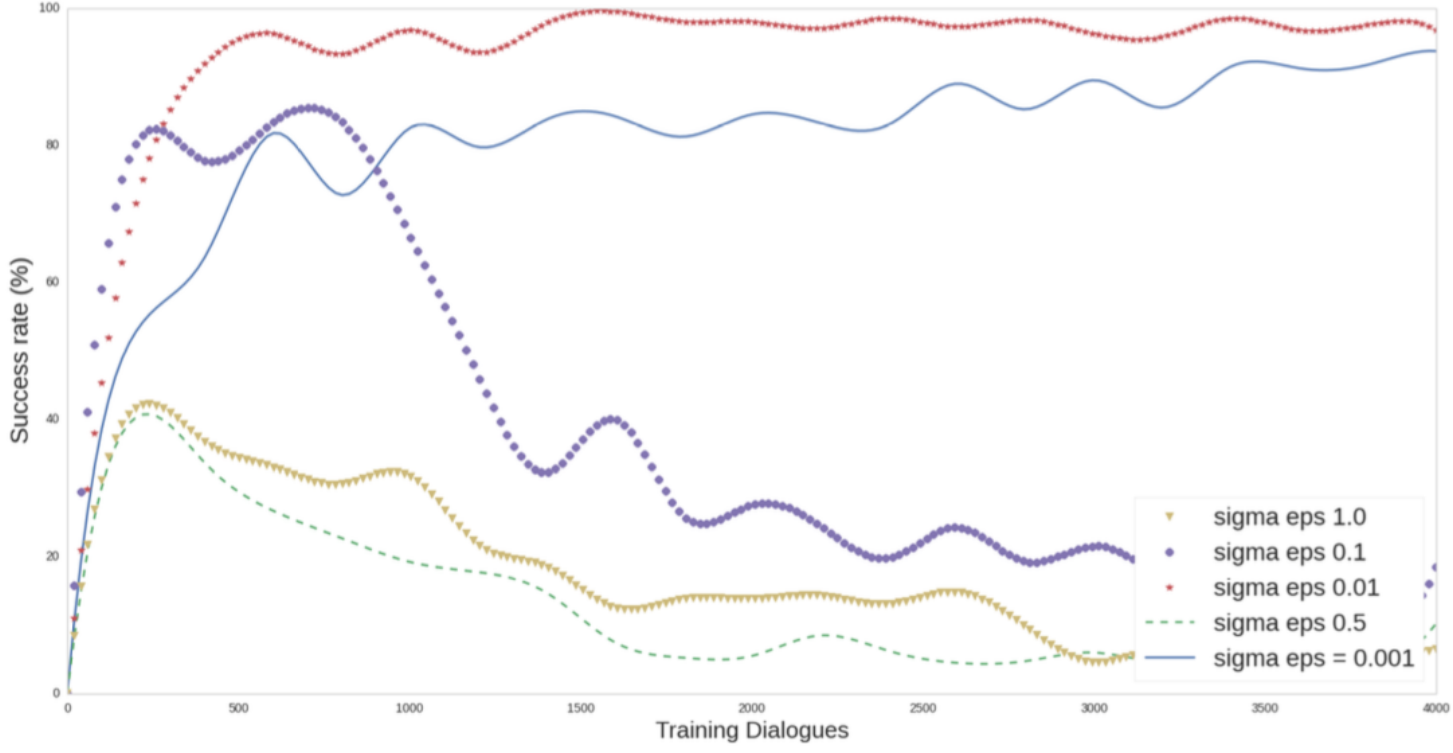
Friendly environment



Results with noise



Robustness of the architecture?



Thank you!

Main References

1. Benchmarking Uncertainty Estimates with Deep Reinforcement Learning for Dialogue Policy Optimisation, ICASSP 2018, Christopher Tegho*, Paweł Budzianowski*, Milica Gasic.
2. Sample-efficient Actor-Critic Reinforcement Learning with Supervised Data for Dialogue Management, SIGDIAL 2017, Pei-Hao Su, Paweł Budzianowski, Stefan Ultes, Milica Gasic and Steve Young.
3. Sample Efficient Deep Reinforcement Learning for Dialogue Systems with Large Action Spaces, Preprint 2018, Gellert Weisz, Paweł Budzianowski, Pei-Hao Su, Milica Gasic.