Dialogue manager domain adaptation using Gaussian process reinforcement learning

Milica Gašić, Nikola Mrkšić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, David Vandyke, Tsung-Hsien Wen and Steve Young

Trumpington Street, Cambridge CB2 1PZ
{mg436,nm480,lmr46,phs26,su259,djv27,thw28,sjy}@cam.ac.uk

Abstract

Spoken dialogue systems allow humans to interact with machines using natural speech. As such, they have many benefits. By using speech as the primary communication medium, a computer interface can facilitate swift, human-like acquisition of information. In recent years, speech interfaces have become ever more popular, as is evident from the rise of personal assistants such as Siri, Google Now, Cortana and Echo. Recently, data-driven machine learning methods have been applied to dialogue modelling and the results achieved for limited-domain applications are comparable to or out-perform traditional approaches. Methods based on Gaussian processes are particularly effective as they enable good models to be estimated from limited training data. Furthermore, they provide an explicit estimate of the uncertainty which is particularly useful for reinforcement learning. This paper explores the additional steps that are necessary to extend these methods to model open domains. We show that Gaussian process reinforcement learning is an elegant framework that naturally supports a range of methods, including prior knowledge, Bayesian committee machines and multi-agent learning, for facilitating extensible and adaptable dialogue systems.

Keywords: Dialogue systems, Reinforcement learning, Gaussian process

1. Introduction

Spoken dialogue systems allow humans to interact with machines using natural speech. As such, they have many benefits. By using speech as the primary communication medium, a computer interface can facilitate swift,

human-like acquisition of information. In recent years, systems with speech interfaces have become ever more popular, as is evident from the rise of personal assistants such as Siri, Google Now, Cortana and Echo. Statistical approaches to dialogue management have been shown to reduce design costs and provide superior performance to hand-crafted systems particularly in noisy environments [1]. Traditionally, spoken dialogue systems were built for limited domains described by an underlying *ontology*, which is essentially a structured representation of the database of entities that the dialogue system can talk about.

The semantic web is an effort to organise the large amount of information available on the Internet into a structure that can be more easily processed by a machine designed to perform reasoning on this data [2]. *Knowledge graphs* are good examples of such structures. They typically consist of a set of triples, where each triple represents two entities connected by a specific relationship. Current knowledge graphs have millions of entities and billions of relations and are constantly growing. There has been a significant amount of work in spoken language understanding focused on exploiting knowledge graphs in order to improve coverage [3, 4]. More recently there have also been some initial attempts to build statistical dialogue systems that operate on large knowledge graphs, but limited so far to the problem of belief tracking [5]. In this paper, we address the problem of decision-making.

Moving from a limited domain dialogue system that operates on a relatively modest ontology size to an open domain dialogue system that can converse about anything in a very large knowledge graph is a non-trivial problem. An open domain dialogue system can be seen as a (large) set of limited domain dialogue systems. If each of them were trained separately then an operational system would require sufficient training data for each individual topic in the knowledge graph, which is simply not feasible. What is more likely is that there will be limited and varied data drawn from different domains. Over time, this data set will grow but there will always be topics within the graph which are rarely visited.

The key to modelling open-domain dialogue systems is therefore the efficient reuse of data. Gaussian processes are a powerful method for efficient function estimation from sparse data. A Gaussian process is a Bayesian method which specifies a prior distribution over the unknown function and then given some observations estimates the posterior [6]. A Gaussian process prior consists of a *mean function* - which is what we expect the unknown function to look like before we have seen any data - and the *kernel function*

which specifies the prior knowledge of the correlation in the data. For every input point, the kernel specifies the expected variation of where the function value will lie and once given some data, the kernel therefore defines the correlations between known and unknown function values. In that way, the known function values influence the regions where we do not have any data points. Also, for every input point the Gaussian process defines a Gaussian distribution over possible function values with mean and variance. When used inside a reinforcement learning framework, the variance can be used to guide exploration, avoiding the need to explore parts of the space where the Gaussian process is very certain. All this leads to very data efficient learning [7].

In this paper, we explore how a Gaussian process-based reinforcement learning framework can be augmented to support open-domain dialogue modelling focussing on three inter-related approaches. The first makes use of the Gaussian process prior. The idea is that where there is little training data available for a specific domain, a *generic* model can be used that has been trained on all available data. Then, when sufficient in-domain data becomes available, the generic model can serve as a prior to build a *specific* model for the given domain. This idea was first proposed in [8].

The second approach is based on a Bayesian committee machine [9]. The idea is that every domain or sub-domain is represented as a committee member. If each committee member is a Bayesian model, e.g. a Gaussian process, then the committee too is a Bayesian model, with mean and variance estimate. If a committee member is trained using limited data its estimates will carry a high uncertainty so the committee will rely on other more confident committee members, until it has seen enough training data. This method was proposed in [10]. It is similar to Products of Gaussians which have previously been applied to problems such as speech recognition [11].

Finally, we extend the committee model to a multi-agent setting where committee members are seen as agents that collaboratively learn. This overarching framework subsumes the first two approaches and provides a practical approach to on-line learning of dialogue decision policies for very large scale systems. It constitutes the primary contribution of this article.

The remainder of the paper is organised as follows. In Section 2, the use of Gaussian process-based reinforcement learning (GPRL) is briefly reviewed. The key advantage of GPRL in this context is that in addition to being data efficient, it directly supports the use of an existing model as a prior thereby facilitating incremental adaptation. In Section 3, various

strategies for building a generic policy are considered and evaluated. We then review the Bayesian committee machine in Section 4.1. Following that, in Section 4.2, we present a multi-domain dialogue manager based on the committee model. In Section 5, we describe how multi-agent learning can be applied to the policy committee model. Then, in Section 6, we present the experimental results. Finally, in Section 7, conclusions together with future research directions are presented.

2. Gaussian process reinforcement learning

The input to a statistical dialogue manager is typically an N-best list of scored hypotheses obtained from the spoken language understanding unit. Based on this input, at every dialogue turn, a distribution of possible dialogue states called the *belief state*, $\boldsymbol{b} \in \mathcal{B}$, an element of *belief space*, is estimated. The belief state must accurately represent everything that happened prior to that turn in the dialogue. The quality of a dialogue is defined by a *reward function* $r(\boldsymbol{b}, a)$ and the role of a dialogue policy π is to map the belief state $\boldsymbol{b}$ into a system action $a \in \mathcal{A}$, an element of *action space*, at each turn so as to maximise the expected cumulative reward.

The expected cumulative reward for a given belief state $\boldsymbol{b}$ and action a is defined by the Q-function:

$$Q(\boldsymbol{b}, a) = E_\pi \left(\sum_{\tau=t+1}^{T} \gamma^{\tau-t-1} r_\tau | b_t = \boldsymbol{b}, a_t = a \right) \tag{1}$$

where r_τ is the immediate reward obtained at time τ, T is the dialogue length and γ is a discount factor, $0 < \gamma \leq 1$. Optimising the Q-function is then equivalent to optimising the policy π.

GP-Sarsa is an on-line reinforcement learning algorithm that models the Q-function as a Gaussian process [12]:

$$Q(\boldsymbol{b}, a) \sim \mathcal{GP}\left(m(\boldsymbol{b}, a), k((\boldsymbol{b}, a), (\boldsymbol{b}, a))\right) \tag{2}$$

where $m(\cdot, \cdot)$ is the prior mean function and the kernel $k(\cdot, \cdot)$ is factored into separate kernels over belief and action spaces $k_{\mathcal{B}}(\boldsymbol{b}, \boldsymbol{b}')k_{\mathcal{A}}(a, a')$.

For a training sequence of belief state-action pairs $\boldsymbol{B} = [(\boldsymbol{b}^0, a^0), \ldots, (\boldsymbol{b}^t, a^t)]^\mathsf{T}$ and the corresponding observed immediate rewards $\boldsymbol{r} = [r^1, \ldots, r^t]^\mathsf{T}$, the posterior of the Q-function for any belief state-action pair $(\boldsymbol{b}, a)$ is given by:

$$Q(\boldsymbol{b}, a) | \boldsymbol{r}, \boldsymbol{B} \sim \mathcal{N}(\overline{Q}(\boldsymbol{b}, a), cov((\boldsymbol{b}, a), (\boldsymbol{b}, a))) \tag{3}$$

where the posterior mean and covariance take the form:

$$\begin{aligned}
&\overline{Q}(\boldsymbol{b}, a) = \boldsymbol{k}(\boldsymbol{b}, a)^{\mathsf{T}} \boldsymbol{H}^{\mathsf{T}} (\boldsymbol{H}\boldsymbol{K}\boldsymbol{H}^{\mathsf{T}} + \sigma^2 \boldsymbol{H}\boldsymbol{H}^{\mathsf{T}})^{-1} (\boldsymbol{r} - \boldsymbol{m}), \\
&cov((\boldsymbol{b}, a), (\boldsymbol{b}, a)) = k((\boldsymbol{b}, a), (\boldsymbol{b}, a)) - \\
&\quad \boldsymbol{k}(\boldsymbol{b}, a)^{\mathsf{T}} \boldsymbol{H}^{\mathsf{T}} (\boldsymbol{H}\boldsymbol{K}\boldsymbol{H}^{\mathsf{T}} + \sigma^2 \boldsymbol{H}\boldsymbol{H}^{\mathsf{T}})^{-1} \boldsymbol{H}\boldsymbol{k}(\boldsymbol{b}, a)
\end{aligned} \tag{4}$$

where $\boldsymbol{m} = [m(\boldsymbol{b}^0, a^0), \ldots, m(\boldsymbol{b}^t, a^t)]^{\mathsf{T}}$, $\boldsymbol{k}(\boldsymbol{b}, a) = [k((\boldsymbol{b}^0, a^0), (\boldsymbol{b}, a)), \ldots, k((\boldsymbol{b}^t, a^t), (\boldsymbol{b}, a))]^{\mathsf{T}}$, $\boldsymbol{K}$ is the Gram matrix [6], $\boldsymbol{H}$ is a band matrix with diagonal $[1, -\gamma]$ and σ^2 is an additive noise factor which controls how much variability in the Q-function estimate is expected during the learning process. Since the Gaussian process for the Q-function defines a Gaussian distribution for every belief state-action pair (3), when a new belief point $\mathbf{b}$ is encountered, for each action $a \in \mathcal{A}$, there is a Gaussian distribution over Q-values. Sampling from these Gaussian distributions gives Q-values $\hat{Q}(\mathbf{b}, a) \sim \mathcal{N}(\overline{Q}(\mathbf{b}, a), \Sigma^Q(\boldsymbol{b}, a))$ where $\Sigma^Q(\boldsymbol{b}, a) = cov((\boldsymbol{b}, a), (\boldsymbol{b}, a))$ from which the action with the highest sampled Q-value can be selected:

$$\pi(\mathbf{b}) = \arg\max_a \left\{ \hat{Q}(\mathbf{b}, a) : a \in \mathcal{A} \right\}. \tag{5}$$

To use GPRL for dialogue, a kernel function must be defined on both the belief state space $\mathcal{B}$ and the action space $\mathcal{A}$. Here we use the Bayesian Update of Dialogue State (BUDS) dialogue model [13]. The action space consists of a set of slot-dependent and slot-independent summary actions which are mapped to master actions using a set of rules and the kernel is defined as:

$$k_{\mathcal{A}}(a, a') = \delta_a(a') \tag{6}$$

where $\delta_a(a') = 1$ iff $a = a'$, 0 otherwise. Slot-dependent summary actions include requesting the slot value, confirming the most likely slot value and selecting between top two slot values. The belief state consists of the probability distributions over the Bayesian network hidden nodes that relate to the dialogue history for each slot and the user goal for each slot. The dialogue history nodes can take a fixed number of values, whereas user goals range over the values defined for that particular slot in the ontology and can have very high cardinalities. User goal distributions are therefore sorted according to the probability assigned to each value since the choice of summary action does not depend on the values but rather on the overall shape of each distribution. The kernel function over both dialogue history and user goal nodes is based on the expected likelihood kernel [14], which is a simple linear

inner product. The kernel function for belief space is then the sum over all the individual hidden node kernels:

$$k_{\mathcal{B}}(\boldsymbol{b}, \boldsymbol{b}') = \sum_h \langle \boldsymbol{b}_h, \boldsymbol{b}'_h \rangle \tag{7}$$

where $\boldsymbol{b}_h$ is the probability distribution encoded in the h^{th} hidden node.

3. Distributed dialogue policies

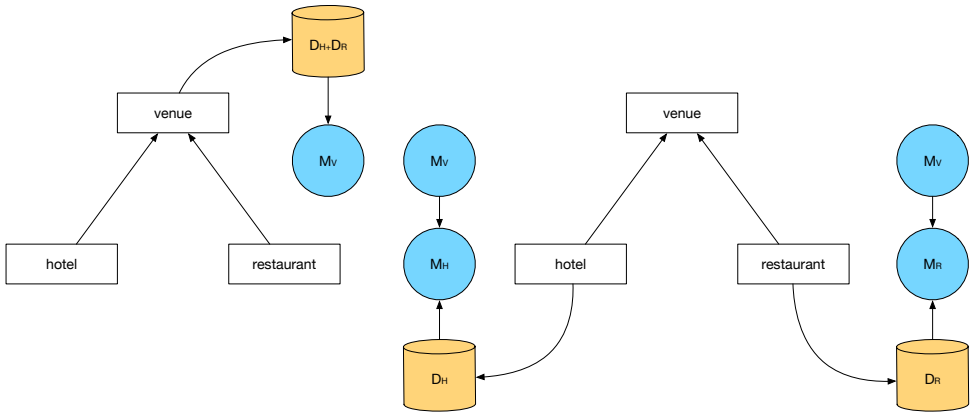

Figure 1: Training a generic venue policy model M_V on data pooled from two subdomains $D_R + D_H$ (left); and training specific policy models M_R and M_H using the generic policy M_V as a prior and additional in-domain training data (right).

One way to build a dialogue manager which can operate across a large knowledge graph is to decompose the dialogue policy into a set of topic specific policies that are distributed across the class nodes in the graph. Initially, there will be relatively little training data and the system will need to rely on generic policies attached to high level generic class nodes which have been trained on whatever examples are available from across the pool of derived classes. As more data is collected, specific policies can be trained for each derived class[1]. An example of this is illustrated in Fig 1. On the left side is the initial situation where conversations about hotels and restaurants are conducted using a generic model M_V trained on example dialogues from both

[1]cf analogy with speech recognition adaptation using regression trees[15]

the hotel and restaurant domains. Once the system has been deployed and more training data has been collected, specific restaurant and hotel models M_R and M_H can be trained.[2]

This type of multi-domain model assumes an agile deployment strategy which can be succinctly described as "deploy, collect data, and refine". Its viability depends on the following assumptions:

1. it is possible to construct generic policies which provide acceptable user performance across a range of differing domains;
2. as sufficient in-domain data becomes available, it is possible to seamlessly adapt the policy to improve performance, without subjecting users to unacceptable disruptions in performance during the adaptation period.

In GPRL, the computation of $Q(\boldsymbol{b}, a)$ requires the kernel function to be evaluated between $(\boldsymbol{b}, a)$ and each of the belief-action points in the training data. If the training data consists of dialogues from subdomains (restaurants and hotels in this case) which have domain-specific slots and actions, a strategy is needed for computing the kernel function between domains.

If domains are organised in a class hierarchy it is expected that they share some of the slots. Calculating the kernel for shared parts of belief state is straightforward:

$$k_{\mathcal{B}}(\boldsymbol{b}^{\mathcal{H}}, \boldsymbol{b}^{\mathcal{R}}) = \sum_{h \in \mathcal{H} \cap \mathcal{R}} \langle \boldsymbol{b}_h^{\mathcal{H}}, \boldsymbol{b}_h^{\mathcal{R}} \rangle, \tag{8}$$

where $\mathcal{R}$ and $\mathcal{H}$ are the considered subdomains. When goal nodes are paired with differing cardinalities (eg name might have different cardinality for different domains), the shorter vector is padded with zeros. The approach used here for non-matching slots is to treat them as *abstract slots* by renaming them as slot-1, slot-2, etc so that they become the same in both subdomains according to some heuristics. Hence for example, food is matched with dogs allowed, and so on. As with the case with shared slots, when goal nodes are paired with differing cardinalities, the shorter vector is padded with zeros. Other adaptation strategies are also possible but may result in increasing the dimensionality (see for example [16]).

[2]Here a model M is assumed to include input mappings for speech understanding, a dialogue policy π and output mappings for generation. In this article, we are only concerned with dialogue management and hence the dialogue policy component π of each model.

4. Committee of dialogue policies

4.1. Bayesian committee machine

The Bayesian committee machine is an approach to combining estimators that have been trained on different datasets. It is particularly suited to Gaussian process regression [9]. Here we apply the method to combine the outputs of multiple estimates of Q-values Q_i with mean $\overline{Q}_i$ and covariance Σ_i^Q as given by Eq. 4. Each estimator is trained on a distinct set of rewards and belief-state action pairs $\boldsymbol{r}_i, \boldsymbol{B}_i$ for $i \in \{1, \ldots, M\}$, where M is the number of policies in the policy committee. As an example, Fig 2 depicts a Bayesian committee machine consisting of three estimators.

Following the description in [9], the combined mean $\overline{Q}$ and covariance Σ^Q are calculated as:

$$\begin{aligned} \overline{Q}(\boldsymbol{b}, a) &= \Sigma^Q(\boldsymbol{b}, a) \textstyle\sum_{i=1}^{M} \Sigma_i^Q(\boldsymbol{b}, a)^{-1} \overline{Q}_i(\boldsymbol{b}, a), \\ \Sigma^Q(\boldsymbol{b}, a)^{-1} &= -(M-1) * k((\boldsymbol{b}, a), (\boldsymbol{b}, a))^{-1} + \textstyle\sum_{i=1}^{M} \Sigma_i^Q(\boldsymbol{b}, a)^{-1}. \end{aligned} \tag{9}$$

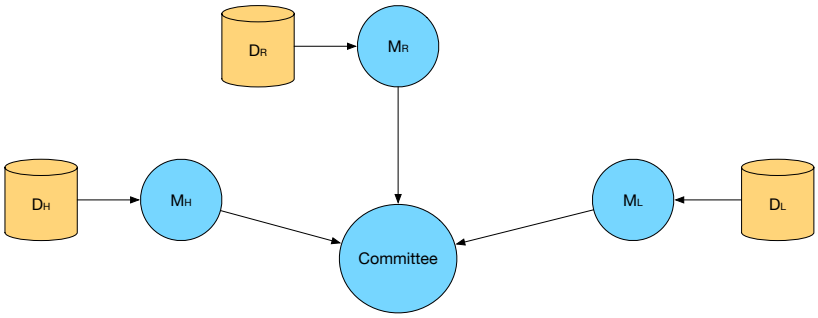

Figure 2: Bayesian committee machine: Committee members consist of estimators trained on different datasets D_i. At every turn their estimated Q-values, Q_i, are combined to determine the final Q-value estimate.

4.2. Multi-domain Dialogue Manager

Section 3 introduced the notion of a *generic* policy, which can be trained from data coming from different domains, and a *specific* policy that can be derived from a generic policy using additional in-domain data. In order to produce a generic policy that works across multiple domains, a kernel

function must be defined on belief states and actions that come from different domains. In this case, domains are organised in a class hierarchy so it is reasonable to assume that there are shared portions of the belief for different domains. These portions relate to shared slots and are directly mapped to each other and for slots which are different, the mapping can be defined either manually or by using some similarity metric.

When using a Bayesian committee machine it is possible to have two domains which have no shared slots. Therefore, a different approach is required for building policies that can operate (and be trained on) belief states and actions that come from different domains. The approach is as follows. The slots from each domain are divided into semantic classes. For instance slot name will be in a different semantic class to slot food. Then, the following steps are taken:

1. For each semantic class and for each slot in that semantic class, the *normalised entropy* η is calculated by

$$\eta(s) = -\sum_{v \in \mathcal{V}_s} \frac{p(s=v)\log(p(s=v))}{|\mathcal{V}_s|}, \tag{10}$$

   where s is a slot that takes values v from a set $\mathcal{V}_s$ and where $p(s=v)$ is the empirical probability that an entity in the database with slot s takes value v for that slot. For example, if all entities in the database for the restaurant domain have area=centre, then that slot has a normalised entropy equal to 0. The measure is normalised so that slots that take different numbers of values can be compared. This measure provides an indicator of how useful each slot is in the dialogue. For instance, in this case it is not useful for the system to ask the user about their preference for slot area since the answer provides no information gain.
2. For each domain, and for each semantic class, the slots are sorted based on their normalised entropy and given abstract names $slot_1^c, slot_2^c, \ldots$ so that $\eta(slot_i^c) \geq \eta(slot_j^c)$ for $i \leq j$ for semantic class c.
3. The kernel function between belief states and actions which come from different domains $\mathcal{M}$ and $\mathcal{N}$ is calculated in the following way:
   - For each $slot_i^c$, $i \leq min(|M_c|, |N_c|)$ in semantic class c where and $|M_c|$ denotes the number of slots in semantic class c in domain M: match the corresponding elements of belief space and actions padding with zeros as necessary.

- Otherwise disregard the elements of belief state relating to unpaired slots j and if one of the actions relates to $slot_j$, consider the action kernel to be 0.

This slot matching process is illustrated in Figure 3.

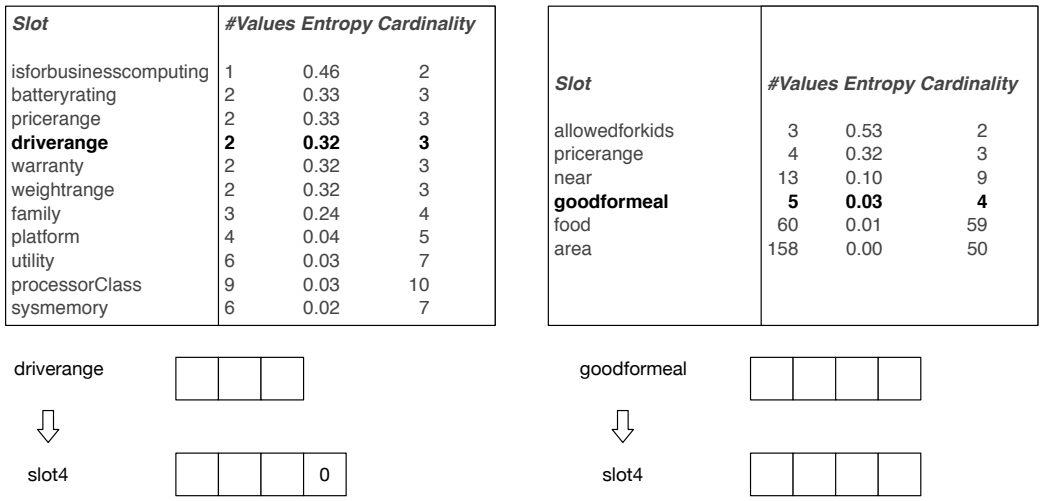

Figure 3: Slot matching for slots that come from different domains.

This approach has three important properties:

1. it does not require human intervention to define the relationship between different domains;
2. it provides a well-defined computable relationship between any two domains; and
3. the kernel function that is defined in step 3 is positive definite so the Gaussian process is well-defined.

5. Multi-agent learning in the policy committee framework

In the standard reinforcement learning framework there is a single agent that is trying to solve a specific task in a given environment. However, for complex tasks it has been shown [17] that it is more effective to decompose the problem into sub-tasks and introduce a distinct agent to solve each subtask. In this case, each agent needs to take into account only part of the state space and this can significantly speed up the learning process. Learning in

such multi-agent systems is typically performed in three steps [17]. First, each agent proposes an action. Second, a gating mechanism, which can be either handcrafted or optimised automatically, is deployed to select the actual system action. Finally, the reward is distributed among the agents and they each re-estimate their policy.

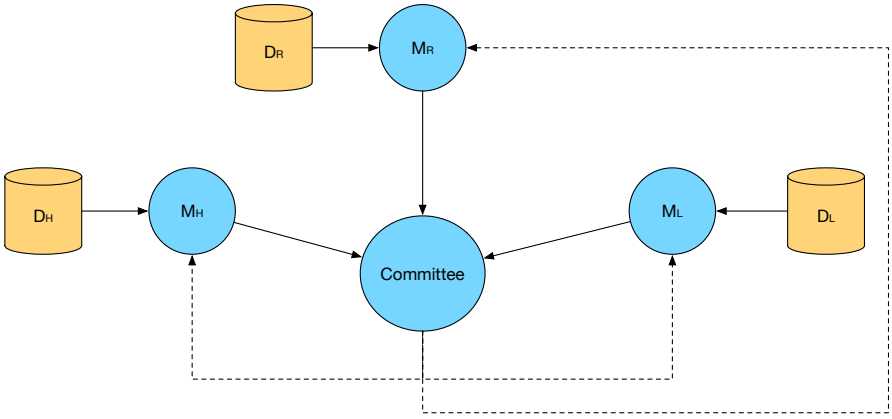

Figure 4: Multi-agent policy committee model

The multi-agent framework can be seen as an extension of the policy committee model (see Figure 4). In fact, the first two steps are exactly the same: each committee member estimates its own Q-function and then Eq. 9 is used as the gating to automatically combine the output. The multi-agent framework, however, includes a third step which is to distribute the reward so that each agent (i.e. committee member) can learn from every dialogue. Intuitively, the reward should be given to the agent for the domain that the dialogue is currently in. However, in practice it can be very difficult to identify a specific domain especially since the user can switch domains within the same dialogue. To avoid these issue, three strategies for distributing the reward are investigated:

naïve approach: the total reward that the system obtains is directly fed back to each committee member [17]

winner-takes-all approach: the total reward that the system obtains is fed back to the committee member that proposed the highest Q-value for the action that was finally chosen by the gating mechanism [18]

reward scaling approach: the total reward is redistributed to each committee member in such a way as to reflect its contribution to the final action chosen by the gating mechanism [17]

6. Experimental results

6.1. Experimental set-up

In order to investigate the effectiveness of the methods discussed above, a variety of contrasts were examined using an agenda-based simulated user operating at the dialogue act level [19, 20]. The reward function allocates -1 at each turn to encourage shorter dialogues, plus 20 at the end of each successful dialogue. The user simulator includes an error generator and this was set to generate incorrect user inputs 15% of time.

The proposed methods were also incorporated into a real-time spoken dialogue system in which policies were trained on-line using subjects recruited via Amazon Mturk. Each user was assigned specific tasks in a given subdomain and then asked to call the system in a similar set-up to that described in [21, 22]. After each dialogue, the users were asked whether they judged the dialogue to be successful or not. Based on that binary rating, the subjective success was calculated as well as the average reward. An objective rating was also computed by comparing the system outputs with the assigned task specification. During training, only dialogues where both objective and subjective score were the same were used.

In order to examine the ability of the proposed methods to operate on multiple domains, four different domains were used:

SFR consisting of restaurants in San Francisco

SFH consisting of hotels in San Francisco

L6 consisting of laptops with 6 properties that the user can specify

L11 same as L6 but with 11 user-specifiable properties.

A description of each domain with slots sorted according to their normalised entropy is given in Table 1.

Table 1: Slots for each domain. The upper half represents slots that can be specified by the user to constrain a search. They are ranked according to normalised entropy (Eq. 10). The remainder are informable slots which can be queried by the user regarding a specific entity.

SFR	SFH	L6	L11
name	name	name	name
allowedforkids	allowedforkids	isforbusiness	isforbusiness
pricerange	pricerange	batteryratings	batteryrating
near	near	pricerange	pricerange
goodformeal	takescreditcards	draverange	draverange
food	hasinternet	weightrange	weightrange
area	area	family	family
-	-	-	platform
-	-	-	utility
-	-	-	processorclass
-	-	-	sysmemory
addr	addr	price	weight
price	phone	drive	battery
phone	postcode	dimension	price
postcode	-	-	dimension
-	-	-	drive
-	-	-	display
-	-	-	graphadaptor
-	-	-	design
-	-	-	processor

6.2. Generic policy performance in simulation

In order to investigate the effectiveness of the generic policies discussed in Section 3, generic policies were trained and then tested in two domains – SFR and SFH using equal numbers of restaurant and hotel dialogues. In addition, in-domain policies were trained as a reference.

For each condition, 10 policies were trained using different random seeds and varying numbers of training dialogues. Each policy was then evaluated using 1000 dialogues in each subdomain. The overall average reward, success rate and number of turns is given in Table 2 together with a 95% confidence interval. The most important measure is the average reward, since the policies are trained to maximise this.

Table 2: Comparison of generic vs in-domain policies. In-domain performance is measured in terms of reward, success rate and the average number of turns per dialog. Results are given with a 95% confidence interval.

Strategy	#Dialogs	Reward	Success	#Turns
SFRestaurant				
in-domain	250	3.26 ± 0.21	60.02 ± 0.97	8.65 ± 0.08
in-domain	500	5.00 ± 0.21	68.17 ± 0.91	8.55 ± 0.07
abstract	500	4.48 ± 0.21	67.35 ± 0.92	8.89 ± 0.08
in-domain	2500	7.95 ± 0.17	83.02 ± 0.75	8.55 ± 0.07
in-domain	5000	8.68 ± 0.15	86.67 ± 0.67	8.54 ± 0.07
abstract	5000	8.58 ± 0.15	86.21 ± 0.68	8.52 ± 0.07
SFHotel				
in-domain	250	3.58 ± 0.21	62.07 ± 0.96	8.75 ± 0.07
in-domain	500	4.83 ± 0.21	69.08 ± 0.92	8.89 ± 0.08
abstract	500	5.27 ± 0.20	70.01 ± 0.90	8.64 ± 0.07
in-domain	2500	8.40 ± 0.16	84.90 ± 0.71	8.46 ± 0.06
in-domain	5000	8.92 ± 0.15	87.48 ± 0.65	8.45 ± 0.06
abstract	5000	8.89 ± 0.15	87.19 ± 0.66	8.44 ± 0.06

As can be seen from Table 2, all generic policies perform better than the in-domain policies trained only on the data available for that subdomain (i.e. half of the training data available for the generic policy in this case) and this is especially the case when training data is limited. This suggests that the provision of generic policies in a large multi-domain will indeed provide

robustness against the user moving into a domain for which there is very little training data.

6.3. Adaptation of in-domain policies using a generic policy as a prior in simulation

We now investigate the effectiveness of using a generic policy as a prior for training an in-domain policy as in the right hand side of Fig. 1. In order to examine the best and worst case, the generic priors (from the 10 randomly seeded examples) that gave the best performance and the worst performance on each sub-domain trained with 500 and 5000 dialogues were selected. This results in four prior policies for each subdomain: abstract-500-worst, abstract-500-best, abstract-5000-worst and abstract-5000-best.

In addition, a policy with no prior was also trained for each subdomain (i.e. the policy was trained from scratch). After every 5000 training dialogues each policy was evaluated with 1000 dialogues. The results are given in Fig. 5 and 6 with a 95% confidence interval. Performance at 0 training dialogues corresponds to using the generic policy as described in the previous section, or using a random policy for the no prior case.

Table 3: Performance of best generic prior when adapted using 50K additional dialogues. Results are given with 95% confidence intervals.

SFR			
Name	Reward	Success	#Turns
best prior	8.66 ± 0.35	85.40 ± 2.19	8.32 ± 0.20
adapted	9.62 ± 0.30	89.60 ± 1.90	8.24 ± 0.19
SFH			
best prior	9.76 ± 0.31	88.80 ± 1.96	7.95 ± 0.21
adapted	10.27 ± 0.27	92.50 ± 1.64	8.20 ± 0.21

The results from Figs. 5 and 6 demonstrate that the performance of the policy in the initial stages of learning are significantly improved using the generic policy as a prior, even if that prior is trained on a small number of dialogues and even if it was the worst performing prior from the batch of 10 training sessions. These results also show that the use of a generic prior does not limit the optimality of the final policy. In fact, the use of a prior can be seen as resetting the variance of a GP which may lead to better sample

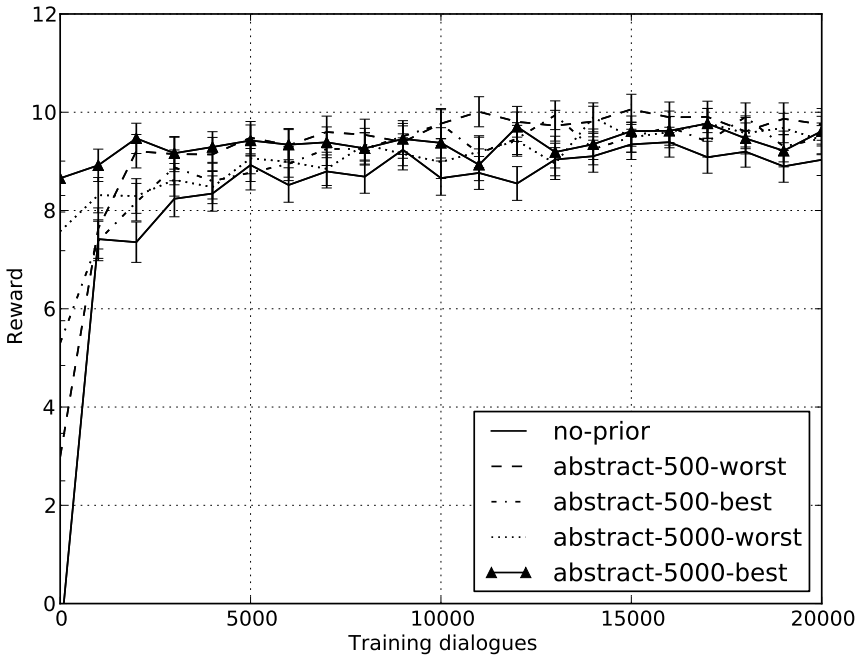

Figure 5: Training policies with different priors – SFR

efficiency [23]. This may be the reason why in some cases the no-prior policies never catch up with the adapted policies.

In Table 3, the performance of the best performing generic prior is compared to the performance of the same policy adapted using an additional 50K dialogues. The results show that additional in-domain adaptation has the potential to improve the performance further. So when enough training data is available, it is beneficial to create individual in-domain policies rather than continuing to train the generic policy. This may be an example of the fact that optimal performance can only be reached when the training and testing conditions match.

6.4. Adaptation in interaction with human users

To examine performance when training with real users rather than a simulator, two training schedules were performed in the SFR subdomain – one training from scratch without a prior and the other performing adaptation using the best generic prior obtained after 5000 simulated training dialogues. For each training schedule three sessions were performed and the results were

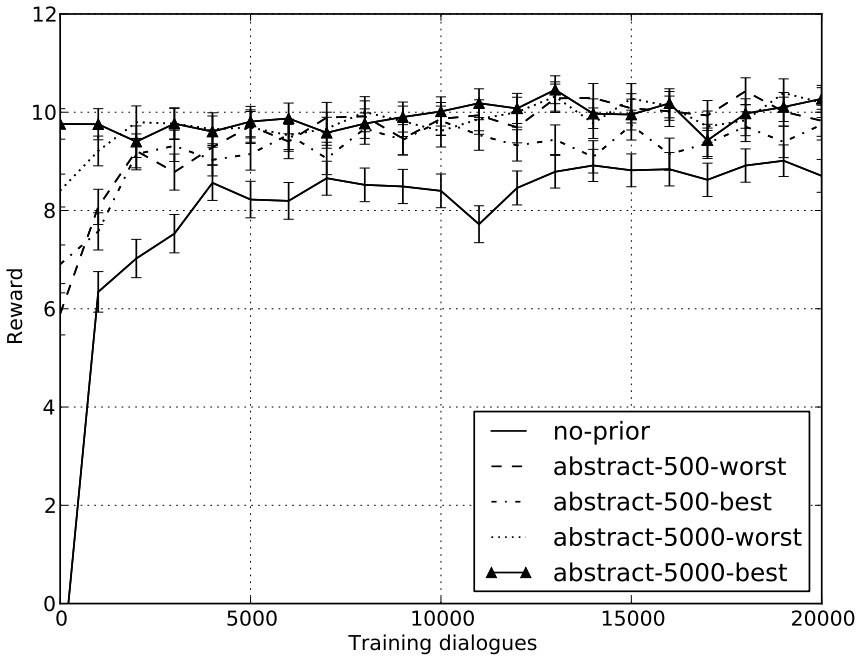

Figure 6: Training policies with different priors – SFH

averaged to reduce any random variation. Fig. 7 shows the moving average reward as a function of the number of training dialogues. The moving window was set to 100 dialogues so that after the initial 100 dialogues each point on the graph is an average of 300 dialogues. The shaded area represents a 95% confidence interval. The initial parts of the graph exhibit more randomness in behaviour because the number of training dialogues is small.

The results show an upward trend in performance particularly for the policy that uses no prior. However, the performance obtained with the prior is significantly better than without a prior both in terms of the reward and the success rate. Equally importantly, unlike the system trained from scratch with no prior, the users of the adapted system are not subjected to poor performance during the early stages of training.

6.5. Policy committee evaluation with simulated user

In the previous section, the benefit of training generic models was demonstrated when training data is sparse. Here we investigate whether the use of a

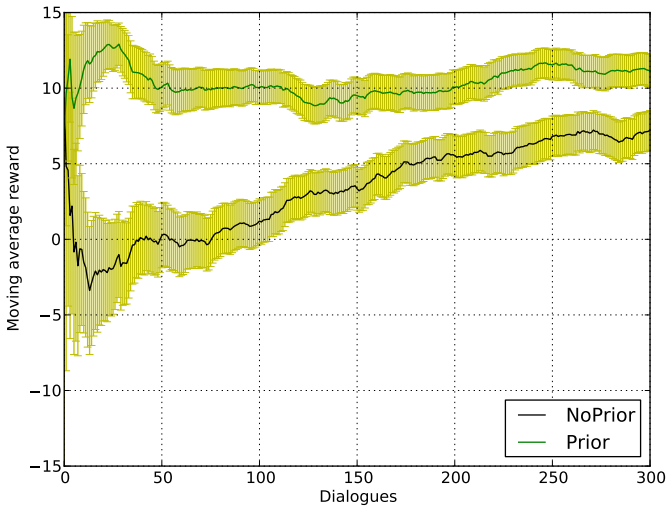

Figure 7: Training in interaction with human users on SFR – moving average reward

Bayesian committee machine can improve robustness further. The contrasts studied were as follows:

INDOM **In-domain policy** – trained only on in-domain data, other data is not taken into consideration, action-selection is based only on the in-domain policy. This is the baseline.

GEN **Single generic policy** – one policy trained on all available data (as in Section 3).

MBCM **Multi-policy Bayesian committee machine** – as described in Section 4.1. There is one committee member for each domain and each committee member is trained only on in-domain data. However, for action-selection, the estimates of all committee members are taken into account using Eq 9, both during training and testing.

GOLD **Gold standard** – this is the performance of the single policy where all training data comes from the same domain i.e. for N domains, GOLD has N times the number of in-domain dialogues for training as provided to INDOM.

We examine two cases: when the training data is limited, with only 250 dialogues available for each domain, and when there is more training data

available, 2500 for each domain. In the evaluation of generic policies in Section 6.2, the test domains were relatively similar. Here, we consider more diverse domains:

- Multi-domain system for SFR, SFH and L6, where the domains have different slots but each domain has the same number of slots, and
- Multi-domain system for SFR, SFH and L11, where not only are the slots different, but also one of the domains, L11, has many more slots than the others.

For each contrast described above, 10 policies were trained on the simulated user using different random seeds. Each policy was then evaluated using 1000 dialogues in each domain. The overall average reward, success rate and number of turns are given in Table 4 together with 95% confidence intervals.

There are several important conclusions to be drawn from the results given in Table 4. First, as shown in Section 6.2, generic policies make use of data that comes from different domains and this improves performance over an in-domain baseline, even in the case presented here where the domains are very different. The multi-policy MBCM results in performance which is either significantly better than other methods or statistically indistinguishable from other methods. In the case of limited training data, its performance is at least as good as the gold standard. Another advantage of MBCM is that it does not require storing a separate generic policy model but only ever produces in-domain models that have the ability to contribute to action-selection for other domains.

Unlike other multi-policy models, MBCM allows flexible selection of committee members. The usefulness of each committee member in the MBCM multi-policy model is explored in Table 5 for the SFR domain. As can be seen from the results, all committee members contribute to performance gains. However not all committee members are equally important. In this case, for good performance on the SFR domain, the SFH committee member is more useful than the L11 committee member.

6.6. Policy committee evaluation with human users

In order to fully examine the effectiveness of the proposed adaptation scheme, policies were also trained in direct interaction with human users.

Table 4: Comparison of strategies for multi-domain adaptation. In-domain performance is measured in terms of reward, success rate and the average number of turns per dialogue. Results are given with 95% confidence intervals.

Strategy	Reward	Success	#Turns
L6 trained on 750 dialogues from SFR, SFH, L6			
INDOM	7.92 ± 0.20	72.64 ± 0.87	6.56 ± 0.07
GEN	9.34 ± 0.19	79.43 ± 0.80	6.49 ± 0.06
MBCM	9.89 ± 0.18	82.95 ± 0.74	6.68 ± 0.07
GOLD	9.25 ± 0.19	80.35 ± 0.79	6.77 ± 0.07
L6 trained on 7500 dialogues from SFR, SFH, L6			
INDOM	10.62 ± 0.16	86.04 ± 0.68	6.50 ± 0.06
MBCM	11.60 ± 0.14	90.32 ± 0.58	6.42 ± 0.06
GOLD	11.98 ± 0.13	92.36 ± 0.53	6.42 ± 0.06
SFR trained on 750 dialogues from SFR, SFH, L11			
INDOM	5.73 ± 0.21	68.17 ± 0.92	7.89 ± 0.08
GEN	6.32 ± 0.21	72.04 ± 0.89	8.05 ± 0.08
MBCM	7.37 ± 0.20	76.60 ± 0.83	7.92 ± 0.08
GOLD	7.34 ± 0.20	76.97 ± 0.83	8.01 ± 0.08
SFR trained on 7500 dialogues from SFR, SFH, L11			
INDOM	9.03 ± 0.17	85.16 ± 0.70	7.97 ± 0.08
MBCM	9.67 ± 0.17	88.28 ± 0.66	7.96 ± 0.08
GOLD	9.65 ± 0.16	88.80 ± 0.62	8.05 ± 0.08
L11 trained on 750 dialogues from SFR, SFH, L11			
INDOM	6.46 ± 0.22	67.59 ± 0.92	7.02 ± 0.08
GEN	7.18 ± 0.21	70.91 ± 0.89	6.97 ± 0.08
MBCM	8.52 ± 0.20	77.09 ± 0.82	6.88 ± 0.07
GOLD	8.68 ± 0.20	77.26 ± 0.83	6.74 ± 0.07
L11 trained on 7500 dialogues from SFR, SFH, L11			
INDOM	10.05 ± 0.17	84.58 ± 0.71	6.84 ± 0.07
MBCM	10.73 ± 0.16	87.23 ± 0.66	6.70 ± 0.07
GOLD	11.17 ± 0.15	88.89 ± 0.62	6.57 ± 0.06

Table 5: Selection of committee members for multi-policy Bayesian committee machine for SFR domain. The committee policy is trained on 7500 dialogues equally spread across three domains.

MBCM – SFR			
Committee members	Reward	Success	#Turns
SFR	7.32 ± 0.22	79.97 ± 0.82	8.51 ± 0.10
SFR+SFH	9.20 ± 0.18	86.51 ± 0.70	8.05 ± 0.09
SFR+L11	8.73 ± 0.19	84.56 ± 0.73	8.12 ± 0.09
SFR+SFH+L11	9.67 ± 0.17	88.28 ± 0.66	7.96 ± 0.08

We compare two set-ups: one where an in-domain L6 policy is trained on-line and another where a multi-policy Bayesian committee machine is trained from scratch using data from the SFR, SFH and L6 domains, which produces a policy committee which can operate on all three domains.

Fig. 8 shows the moving average reward as a function of the number of training dialogues for the L6 domain comparing the in-domain (INDOM) policy and the multi-policy Bayesian committee machine (MBCM) as defined in Section 4.2. The performance of the MBCM policy was only shown on training dialogues that came from the L6 domain, but in fact it was also trained on SFR and SFH domains in parallel. The training data across the domains was equally distributed. The moving window was set to 100 dialogues so that after the initial 100 dialogues each point on the graph is an average of 300 dialogues. The shaded area represents a 95% confidence interval. The initial parts of the graph exhibit more randomness in behaviour because the number of training dialogues is small. The results show that the multi-policy Bayesian committee machine consistently outperforms the in-domain policy. To the best of our knowledge, this is the first time a dialogue policy has been trained on multiple domains on-line in interaction with real users.

6.7. Multi-agent simulation results

Finally, we examine the effectiveness of extending the policy committee model to multi-agent learning. The contrasts studied were as follows:

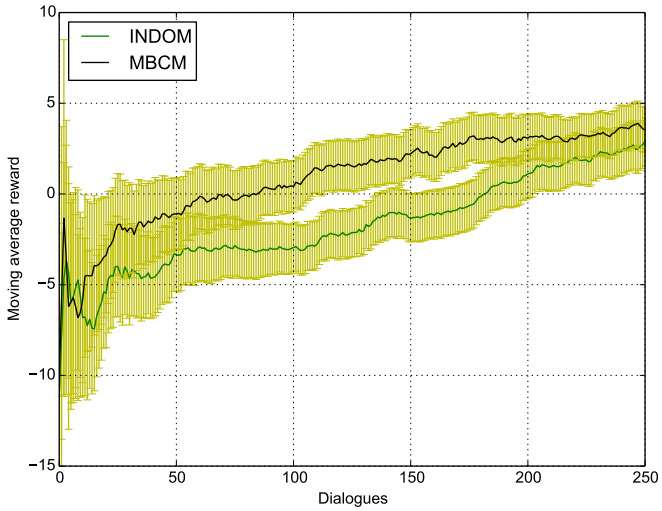

Figure 8: Training in interaction with human users on L6 domain – moving average reward

NAÏV **Naïve approach** – The total reward is given to each committee member during every interaction regardless of the current domain.

WINN **Winner-takes-all approach** – The total reward is given to the policy member which on average gave the highest Q-value Q-variance ratio during the whole dialogue, $\Sigma_i^Q(\boldsymbol{b}, a)^{-1}\overline{Q}_i(\boldsymbol{b}, a)$ from Eq. 9, for the last action taken by the system.

SCALE **Scale received reward according to all committee members' Q-value estimate** – the total reward is distributed to each policy committee member in proportion to the average Q-value Q-variance ratio, $\Sigma_i^Q(\boldsymbol{b}, a)^{-1}\overline{Q}_i(\boldsymbol{b}, a)$ from Eq. 9, for the last action that the system took relative to the Q-value Q-variance ratios of the other committee members for the last action.

MBCM **Multi-policy Bayesian committee machine** – Each committee member is trained only on in-domain data, so the reward is passed only to the committee member which is specific to that domain (see Section 4.2 for details).

We consider a multi-domain system for SFR, SFH and L11. Two scenarios are examined: (a) when the training data is limited, with only 250 dialogues

available for each domain, and (b) when there is more training data available, 2500 for each domain.

For each method described above, 10 policies were trained on the simulated user using different random seeds. Each policy was then evaluated using 1000 dialogues on each domain. The overall average reward, success rate and number of turns are given in Table 6 together with 95% confidence intervals.

There are some important conclusions to be drawn from these results. First, on a smaller dataset the WINN approach which chooses the winning committee member to pass the total reward to is less effective than the approaches which distribute the reward. This is expected, as in the latter case the policy learns from a larger set of dialogues, which is particularly useful in the early stages of the optimisation process. On larger datasets, the winner-takes-all approach gives similar or better performance to the approaches which distribute the reward. Again, this is in line with the intuition that with abundant data, the accuracy of the given reward is more important than the amount of training data. If we average results across the domains and the sizes of the training data, however, we can see that it is generally more effective to use the approaches which distribute the reward.

It is also important to understand behaviour when a new domain is added alongside a set of existing agents which themselves are not yet fully trained. We are interested in both the performance in the new domain as well as the existing domains. To investigate this, two agents operating in the SFR and SFH domains were pre-trained with 250 dialogues each using SCALE reword distribution mechanism. Performance was then evaluated in the SFR and the new as yet untrained L11 domain. The L11 agent was then trained with 250 dialogues in that domain. Again, the performance was tested in both L11 and SFR. Finally training continued with another 250 dialogues for each of the three domains - SFR, SFH and L11 and the performance in the SFR and L11 domains tested for a final time. The results are shown in Table 7.

The performance of the dialogue manager in the L11 domain when trained only with SFR and SFH dialogues is very poor, which is expected as these are very different domains. However with the addition of 250 L11 dialogues, the performance dramatically improves. What is more, adding these L11 dialogues does not impede performance in the SFR domain, in fact it improves slightly. With an additional 750 dialogues spread across all three domains, the performance significantly improves in both the L11 and SFR domains.

Table 6: Comparison of strategies for multi-domain adaptation. In-domain performance is measured in terms of reward, success rate and the average number of turns per dialogue. Results are given with 95% confidence intervals.

Strategy	Reward	Success	#Turns
SFR trained on 750 dialogues from SFR, SFH, L11			
NAÏV	7.00 ± 0.20	73.66 ± 0.86	7.70 ± 0.08
WINN	6.84 ± 0.21	75.81 ± 0.84	8.29 ± 0.09
SCALE	7.06 ± 0.21	75.29 ± 0.85	7.98 ± 0.09
MBCM	7.37 ± 0.20	76.60 ± 0.83	7.92 ± 0.08
L11 trained on 750 dialogues from SFR, SFH, L11			
NAÏV	8.82 ± 0.20	77.40 ± 0.82	6.63 ± 0.07
WINN	7.23 ± 0.22	72.35 ± 0.88	7.20 ± 0.09
SCALE	8.11 ± 0.21	74.61 ± 0.85	6.78 ± 0.08
MBCM	8.52 ± 0.20	77.09 ± 0.82	6.88 ± 0.07
SFR trained on 7500 dialogues from SFR, SFH, L11			
NAÏV	9.45 ± 0.22	87.98 ± 0.85	8.14 ± 0.11
WINN	9.67 ± 0.18	89.24 ± 0.68	8.15 ± 0.09
SCALE	9.41 ± 0.17	88.08 ± 0.66	8.18 ± 0.09
MBCM	9.67 ± 0.17	88.28 ± 0.66	7.96 ± 0.08
L11 trained on 7500 dialogues from SFR, SFH, L11			
NAÏV	10.92 ± 0.16	86.80 ± 0.70	6.42 ± 0.07
WINN	11.25 ± 0.18	88.51 ± 0.76	6.43 ± 0.08
SCALE	11.24 ± 0.17	88.55 ± 0.69	6.44 ± 0.07
MBCM	10.73 ± 0.16	87.23 ± 0.66	6.70 ± 0.07
Averaged across domains and size of training data			
NAÏV	8.94 ± 0.10	80.49 ± 0.42	7.13 ± 0.04
WINN	8.46 ± 0.10	80.37 ± 0.42	7.58 ± 0.05
SCALE	8.83 ± 0.10	81.17 ± 0.40	7.38 ± 0.04
MBCM	9.06 ± 0.09	82.17 ± 0.38	7.35 ± 0.04

Table 7: Performance when adding a new agent for the L11 domain to a multi-domain dialogue manager using SCALE training with two partially trained agents for the SFR and SFH domains.

Performance in L11 domain			
Training data	Reward	Success	Turns
250 SFR+ 250 SFH	-10.89 ± 0.40	39.89 ± 0.96	16.65 ± 0.21
+250 L11	4.18 ± 0.28	62.18 ± 0.95	7.89 ± 0.11
+250 SFR+250 SFH+250 L11	7.26 ± 0.22	70.47 ± 0.94	6.79 ± 0.08
Performance in SFR domain			
250 SFR+ 250 SFH	6.12 ± 0.22	70.22 ± 0.91	7.90 ± 0.08
+250 L11	6.75 ± 0.21	73.54 ± 0.86	7.93 ± 0.08
+250 SFR+250 SFH+250 L11	8.05 ± 0.20	79.38 ± 0.83	7.79 ± 0.08

6.8. Multi-agent human user evaluation

To ensure that the benefits of the proposed reward distribution approach suggested by the above simulation results carry over into systems trained online, two systems were also trained in direct interaction with human users. First, a multi-policy Bayesian committee machine (MBCM) was trained from scratch using data from the SFR restaurant, the SFH hotel and the L6 laptop domains. This MBCM policy committee machine operates on all three domains but is dependent on the knowledge of the current domain for policy updating. This is compared to the committee reward scaling (SCALE) machine, presented in Section 6.7, which distributes the reward to every committee member for each dialogue regardless of the domain. The system was deployed in a telephone-based set-up, with subjects recruited via Amazon MTurk and a recurrent neural network model was used to estimate the reward [24].

Fig. 9 shows the moving average reward as a function of the number of training dialogues for the L6 domain comparing the MBCM and SCALE committee approaches. The committees were also trained on SFR and SFH domains in parallel. The training data across the domains was equally distributed. The moving window was set to 100 dialogues so that after the initial 100 dialogues each point on the graph is an average of 300 dialogues. The shaded area represents a 95% confidence interval. As can be seen from

the reward graph for the SCALE approach, the results confirm that it is not necessary for the committee to be aware of the domain. On the contrary, distributing reward to each committee member according to their contribution can even produce better performance than only sending the reward signal to the committee member dedicated to the current domain.

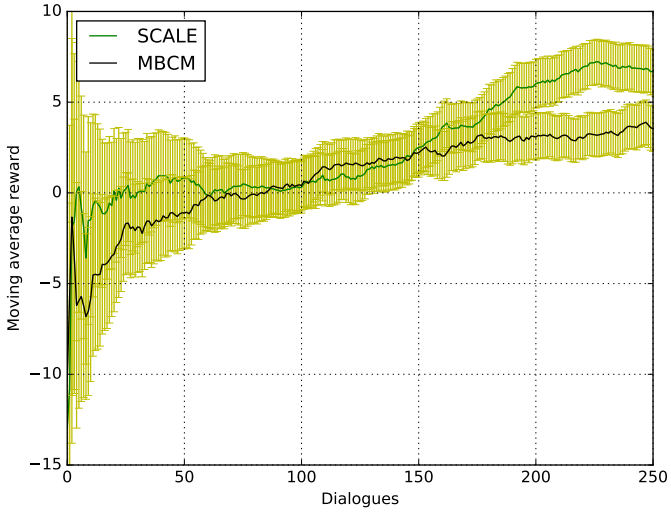

Figure 9: Training using the MBCM and SCALE approaches in interaction with human users in the L6 laptop domain – moving average reward

7. Conclusion

This paper has described three models which support dialogue system domain extension. First, a distributed multi-domain dialogue architecture was proposed in which dialogue policies are organised in a class hierarchy aligned to an underlying knowledge graph. The class hierarchy allows a system to be deployed by using a modest amount of data to train a small set of generic policies. As further data is collected, generic policies can be adapted to give in-domain performance. Using Gaussian process-based reinforcement learning, it has been shown that it is possible to construct generic policies which provide acceptable in-domain user performance, and better performance than can be obtained using under-trained domain specific policies. To construct a generic policy, a design consisting of all common slots plus a number of abstract slots which can be mapped to domain-specific slots works well. It

has also been shown that as sufficient in-domain data becomes available, it is possible to seamlessly adapt to improve performance, without subjecting users to unacceptable disruptions in performance during the adaptation period and without limiting the final performance compared to policies trained from scratch.

An alternative to hierarchically structured policies is the distributed committee model which uses estimates from different policies for action selection at every dialogue turn. The results presented have shown that this model is particularly useful for training multi-domain dialogue systems where the data is limited and varied. As shown in both simulations and in real user trials, the Bayesian policy committee approach gives superior performance to the traditional one-policy-approach across multiple domains and allows flexible selection of committee members during testing.

Finally, the basic policy committee model was extended using ideas from multi-agent learning to distribute the reward signal among the committee members. This model is particularly useful in real-world scenarios where the domain is a priori unknown and indeed, may change during a dialogue. In simulations, the proposed approach achieves a performance which is close to that which relies on explicit domain information to assign reward, while in a real human trial, it produced better performance.

For future work, these methods will be applied to a dialogue manager operating over a large knowledge graph in order to demonstrate that they do indeed scale and offer a viable approach to building truly open domain spoken dialogue systems which learn on-line in interaction with real users.

8. Acknowledgements

The research leading to this work was funded by the EPSRC grant EP/M018946/1 "Open Domain Statistical Spoken Dialogue Systems".

[1] S. Young, M. Gašić, B. Thomson, J. Williams, Pomdp-based statistical spoken dialogue systems: a review, Proceedings IEEE 101 (5) (2013) 1160–1179.

[2] P. Szeredi, G. Lukácsy, T. Benkö, The Semantic Web Explained: The Technology and Mathematics Behind Web 3.0, Cambridge University Press, New York, NY, USA, 2014.

[3] G. Tür, M. Jeong, Y.-Y. Wang, D. Hakkani-Tür, L. P. Heck, Exploiting the semantic web for unsupervised natural language semantic parsing, in: Proceedings of Interspeech, 2012.

[4] L. P. Heck, D. Hakkani-Tür, G. Tür, Leveraging knowledge graphs for web-scale unsupervised semantic parsing., in: Proceedings of Interspeech, 2013, pp. 1594–1598.

[5] Y. Ma, P. A. Crook, R. Sarikaya, E. Fosler-Lussier, Knowledge graph inference for spoken dialog systems, in: Proceedings of 40th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2015, IEEE Institute of Electrical and Electronics Engineers, 2015.
URL *http://research.microsoft.com/apps/pubs/default.aspx?id=240846*

[6] C. Rasmussen, C. Williams, Gaussian Processes for Machine Learning, MIT Press, Cambridge, Massachusetts, 2005.

[7] M. Gašić, S. Young, Gaussian Processes for POMDP-Based Dialogue Manager Optimization, TASLP 22 (1).

[8] M. Gašić, D. Kim, P. Tsiakoulis, S. Young, Distributed dialogue policies for multi-domain statistical dialogue management, in: Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on, IEEE, 2015, pp. 5371–5375.

[9] V. Tresp, A Bayesian Committee Machine, Neural Comput. 12 (11) (2000) 2719–2741. doi:10.1162/089976600300014908.
URL *http://dx.doi.org/10.1162/089976600300014908*

[10] M. Gašić, N. Mrkšić, P.-H. Su, D. Vandyke, T.-H. Wen, S. Young, Policy committee for adaptation in multi-domain spoken dialogue systems, in: Proceedings of ASRU, 2015.

[11] M. J. F. Gales, S. S. Airey, Product of gaussians for speech recognition, Comput. Speech Lang. 20 (1) (2006) 22–40. doi:10.1016/j.csl.2004.12.002.
URL *http://dx.doi.org/10.1016/j.csl.2004.12.002*

[12] Y. Engel, S. Mannor, R. Meir, Reinforcement learning with Gaussian processes, in: Proceedings of ICML, 2005.

[13] B. Thomson, S. Young, Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems, Computer Speech and Language 24 (4) (2010) 562–588.

[14] T. Jebara, R. Kondor, A. Howard, Probability product kernels, J. Mach. Learn. Res. 5 (2004) 819–844.

[15] M. F. Gales, The Generation And Use Of Regression Class Trees For MLLR Adaptation, Tech. Rep. CUED/F-INFENG/TR.263, Cambridge University Engineering Dept (1996).

[16] H. Daume III, Frustratingly easy domain adaptation, in: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, Association for Computational Linguistics, Prague, Czech Republic, 2007, pp. 256–263.

[17] P. Raicevic, Parallel reinforcement learning using multiple reward signals, Neurocomputing 69 (1618) (2006) 2171 – 2179.

[18] M. Humphrys, W-learning: competition among selfish Q-learners, Tech. Rep. UCAM-CL-TR-362, University of Cambridge, Computer Laboratory (Apr. 1995).
URL *http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-362.ps.gz*

[19] J. Schatzmann, B. Thomson, K. Weilhammer, H. Ye, S. Young, Agenda-Based User Simulation for Bootstrapping a POMDP Dialogue System, in: Proceedings of HLT, 2007.

[20] S. Keizer, M. Gašić, F. Jurčíček, F. Mairesse, B. Thomson, K. Yu, S. Young, Parameter estimation for agenda-based user simulation, in: Proceedings of SIGDIAL, 2010.

[21] F. Jurčíček, S. Keizer, M. Gašić, F. Mairesse, B. Thomson, K. Yu, S. Young, Real user evaluation of spoken dialogue systems using Amazon Mechanical Turk, in: Proceedings of Interspeech, 2011.

[22] M. Gašić, C. Breslin, M. Henderson, M. Szummer, B. Thomson, P. Tsiakoulis, S. Young, On-line policy optimisation of Bayesian Dialogue Systems by human interaction, in: Proceedings of ICASSP, 2013.

[23] R. C. Grande, T. J. Walsh, J. P. How, Sample efficient reinforcement learning with gaussian processes, in: Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014, 2014, pp. 1332–1340.

[24] P.-H. Su, D. Vandyke, M. Gašić, D. Kim, N. Mrkšić, T.-H. Wen, S. Young, Learning from real users: Rating dialogue success with neural networks for reinforcement learning in spoken dialogue systems, in: Proceedings of Interspeech, 2015.