Autonomous Sub-domain Modeling for Dialogue Policy with Hierarchical Deep Reinforcement Learning

Giovanni Yoko Kristianto[1], Huiwen Zhang[2, 3], Bin Tong[1],
Makoto Iwayama[1], Yoshiyuki Kobayashi[1]

[1]Hitachi Central Research Laboratory, Tokyo, Japan
[2]Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China
[3]University of Chinese Academy of Sciences, Beijing, China
{*yokogiovanni.kristianto.oq, bin.tong.hh, makoto.iwayama.nw, yoshiyuki.kobayashi.gp*}*@hitachi.com*
zhanghuiwen@sia.cn

Abstract

Solving composites tasks, which consist of several inherent sub-tasks, remains a challenge in the research area of dialogue. Current studies have tackled this issue by manually decomposing the composite tasks into several sub-domains. However, much human effort is inevitable. This paper proposes a dialogue framework that autonomously models meaningful sub-domains and learns the policy over them. Our experiments show that our framework outperforms the baseline without sub-domains by 11% in terms of success rate, and is competitive with that with manually defined sub-domains.

1 Introduction

Modeling a composite dialogue (Peng et al., 2017), which consists of several inherent sub-tasks, is in high demand due to the complexity of human conversation. For instance, a composite dialogue of making a hotel reservation involves several sub-tasks, such as looking for a hotel that meets the user's constraints, booking the room, and paying for the room. The completion of a composite dialogue requires the fulfillment of all involved sub-tasks. In this paper, we focus on the development of a dialogue agent that can discover inherent sub-tasks autonomously from a composite domain, learn a policy to fulfill each sub-task, and learn a policy among these sub-tasks to solve the composite task. Composite dialogues are different from multi-domain dialogues. In multi-domain dialogue systems (Cuayáhuitl et al., 2016; Gasic et al., 2016), each dialogue typically involves one domain, and consequently, its fulfillment does not need policy across domains.

To develop a dialogue agent that can handle a composite task, using standard flat reinforcement learning (RL), which are often used for dialogues with a simple task (Young et al., 2013; Gašić and Young, 2014; Williams et al., 2017; Casanueva et al., 2017; Li et al., 2017), might be inappropriate. Flat RL methods, such as DQN (Mnih et al., 2015), could suffer from the curse of dimensionality, that is the number of parameters to be learned grows exponentially with the size of any compact encoding of system state. Therefore, flat RL is unable to learn reliable value functions (Kulkarni et al., 2016) for a composite task. A composite task has a larger state space and action set, longer trajectory, and more sparse rewards than a simple task. Hierarchical reinforcement learning (HRL) (Dietterich, 2000; Parr and Russell, 1997) is a technique to model complex dialogues (Cuayáhuitl, 2009). Peng et al. (2017) and Budzianowski et al. (2017) used the options framework (Sutton et al., 1999) to solve the above problems in composite dialogues and showed its superiority over flat RL. In their work, however, each option (i.e. sub-task) and its property (e.g. starting and terminating conditions, and valid action set) had to be manually defined. Such handcrafted options ease the policy learning in a composite task, but much human effort is inevitable.

To solve the above problems, we propose to model sub-domains autonomously without any human intervention. The modeled sub-domains imitate the intentions to fulfill sub-tasks in a dialogue, which consequently can be reused by similar yet different domains. Challenges to achieve such autonomous sub-domain modeling include (i) how to discover meaningful sub-domains and their properties (i.e. starting conditions, terminating conditions, and the policies), and (ii) how to have a coherent interaction among these sub-domains so that the dialogue agent can accomplish a dialogue goal efficiently. To tackle these challenges, we propose a unified framework that integrates *option discovery* (Bacon et al., 2017; Machado et al., 2017) with HRL to learn the opti-

Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd Int'l Workshop on Search-Oriented Conversational AI, pages 9–16
Brussels, Belgium, October 31, 2018.
ISBN 978-1-948087-75-9

mal policies over options. With an evaluation involving a task of reserving hotel room, we confirm that our framework achieves a significant improvement over flat RL by 11% in terms of success rate, and is competitive with the framework with manually defined options (Budzianowski et al., 2017).

2 Hierarchical Policy Management

A composite task can be decomposed into a sequence of sub-domains, which are also called options. The composite task is accomplished when all these sub-domains are fulfilled. Following the options framework (Sutton et al., 1999), our dialogue agent handles the composite task by designing two levels of policies in a hierarchical structure, as shown in Figure 1.

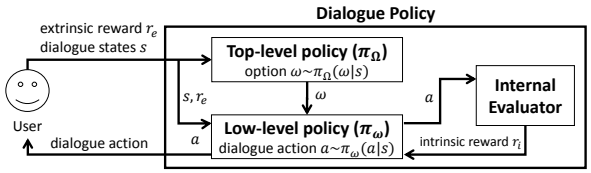


Figure 1: Overview of our dialogue policy.

In this hierarchical policy framework, $\mathcal{S}$ denotes the dialogue state space, Ω the option space, and $\mathcal{A}$ the action set. For a dialogue state $s \in \mathcal{S}$, the top-level policy π_Ω determines which option $\omega \in \Omega$ should be chosen. Then, the policy π_ω determines which primitive action $a \in \mathcal{A}$ should be chosen in option ω for s. As shown by the example in Figure 2, a primitive action is an action lasting for one time step, while an option is an action lasting several time steps. For each s, a dialogue action, which is a primitive action, is returned to the user. The dialogue system will receive an extrinsic reward r_e and a new belief state s'. An optimal policy π^* maximizes the expected discounted return $G_t = \mathbb{E}_{\pi,P}[\sum_{k=0}^{\infty} \gamma^k r_{e,t+k+1} | s_t]$ at every time step t, where P is a transition probability kernel, $\gamma \in [0, 1]$ is a discount factor, and $r_{e,t'}$ is the extrinsic reward obtained at step t'.

Figure 2 shows an example of the execution of our hierarchical dialogue policy in a dialogue domain about hotel room reservation. This domain comprises two sub-domains, i.e., searching for a hotel and booking a hotel room. In this example, we assume that the dialogue system has prior knowledge regarding these sub-domains. In this paper, we propose a dialogue framework that can autonomously discover such sub-domains.

3 Autonomous Sub-Domain Modeling

An option is defined as 3-tuple $\omega = \langle \mathcal{I}_\omega, \pi_\omega, \beta_\omega \rangle$, where $\mathcal{I}_\omega \subseteq \mathcal{S}$ is the initiation set of states where ω can be chosen, $\pi_\omega : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ is the policy of ω, and $\beta : \mathcal{S} \rightarrow [0, 1]$ is the termination condition of ω. To autonomously discover options and learn their policies, we proposed to integrate option-critic (OC) (Bacon et al., 2017) and proto-value functions (PVFs) (Mahadevan, 2007; Machado et al., 2017) into a unified framework.

3.1 Option-Critic Architecture

OC is a gradient-based approach for simultaneously learning intra-option policies π_ω and termination functions β_ω. It learns options gradually from its interactions with environment. It uses option value function $Q_\Omega(s, \omega)$ defined as follows.

$$Q_\Omega(s, \omega) = \sum_a \pi_\omega(a|s) Q_U(s, \omega, a)$$

$$Q_U(s, \omega, a) = r(s, a) + \gamma \sum_{s'} P(s'|s, a) U(\omega, s')$$

$$U(\omega, s') = (1 - \beta_\omega(s')) Q_\Omega(s', \omega) + \beta_\omega(s') V_\Omega(s')$$

$Q_U(s, \omega, a)$ is the value of executing an action in the context of a state-option pair, and $U(\omega, s')$ is the utility from s' onwards, given that we arrive in s' using ω. We parameterize π_ω by θ and β_ω by ϑ. The learning algorithm of OC involves two steps:

- *options evaluation:* updating Q_Ω and Q_U with temporal difference errors; and
- *options improvement:* updating θ with $\frac{\partial Q_\Omega}{\partial \theta}$ and ϑ with $\frac{\partial Q_\Omega}{\partial \vartheta}$.

To obtain policy π_Ω over options, we combine OC with intra-option Q-learning (Sutton et al., 1999). Hereinafter, this combination is denoted as HRL-OC.

HRL-OC optimizes the options and their policies for maximizing the cumulative *extrinsic* reward. It is focused less on discovering meaningful options (Bacon et al., 2017), which may result in unnatural sub-domains in a successful conversation. To tackle this issue, we use PVFs, which are capable of capturing the geometry of the state space, to discover meaningful sub-domains.

3.2 Proto-Value Functions as Options

Proto-value functions (PVFs) are learned representations that approximate state-value function in RL (Mahadevan, 2007). Machado et al. (2017)

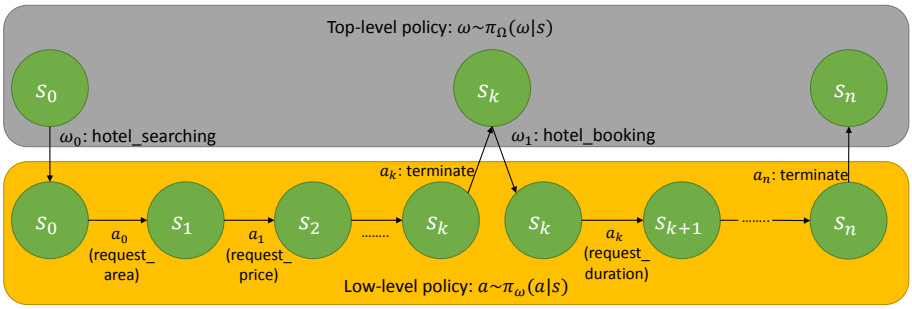


Figure 2: An example of the execution of our hierarchical dialogue policy in hotel reservation domain. At time $t = 0$ and $t = k$, given the belief state s_t, top-level policy π_Ω takes options ω_0 and ω_1, respectively. ω_0 lasts for k turns until its policy π_{ω_i} takes terminate action, while ω_1 lasts for $n - k$ turns.

further demonstrated that PVFs implicitly define options. PVF-based option discovery extracts options from the topology structure of the state space and is capable of providing dense *intrinsic* rewards for each option. The discovery process is given below.

Given a set of sampled state transitions, we construct an adjacency matrix W between belief states using Gaussian kernel. Then, we apply eigendecomposition to the combinatorial graph Laplacian of W. Each eigenvector (i.e. PVF) $\boldsymbol{e}_\omega$ corresponds to an option with intrinsic reward function $r_i^\omega(s, s') = \boldsymbol{e}_\omega[s'] - \boldsymbol{e}_\omega[s]$ for a state transition from s to s'. Since our dialogue system has continuous belief states, we interpolate the value of eigenvectors to novel states using Nyström approximation (Mahadevan, 2007). The number of generated intrinsic reward functions is equal to the number of dialogue states in W, but we used intrinsic reward functions from eigenvectors with the smallest eigenvalues.

An option ω, which corresponds to an eigenvector $\boldsymbol{e}_\omega$, can be interpreted as a desire to reach a belief state s that has the highest value of $\boldsymbol{e}_\omega[s]$ (Machado et al., 2017). In our experiment, such a state usually represents a dialogue goal or a state where user's inherent sub-domain changes (e.g. user starts the booking sub-domain once she finds the hotel satisfying her requirements).

3.3 Policy Learning with Intrinsic Rewards

To realize a dialogue framework that can discover effective and meaningful sub-domains, we feed PVFs into HRL-OC, then follows HRL-OC's learning procedure. Here, PVFs act as an internal evaluator of the dialogue policy. We formulate the $r(s, a)$ in Q_U to be $r(s, a) = \alpha r_i^\omega + (1 - \alpha) r_e$. Hereinafter, this model is denoted as HRL-OC_PVF. We can regard HRL-OC as HRL-OC_PVF with $\alpha = 0$.

We also introduce alternative dialogue frameworks by applying the intrinsic rewards from PVFs directly to HRL algorithms. We train each policy π_ω in HRL using a specific intrinsic reward function r_i^ω. We implemented the hierarchical deep Q-networks (HRL-DQN; Kulkarni et al. (2016)), and policy gradient-DQN (HRL-PG_DQN), i.e., REINFORCE (Williams, 1992) as the top-level policy and DQN low-level policy. This assesses whether using only general-purpose intrinsic rewards, which are designed for exploration, is good for maximizing extrinsic rewards.

4 Experimental Setup

We conducted three evaluations on (i) the effectiveness of our autonomous sub-domain modeling compared to the manual sub-domain modeling, (ii) the performance difference between flat RL (i.e. without modeling) and the HRL with autonomous modeling, and (iii) the impact of using PVFs in discovering meaningful sub-domains.

4.1 Dialogue Domain

Following the setting in Budzianowski et al. (2017), we evaluated our proposed framework in the task of reserving a hotel room, which involves three sub-domains: searching for a hotel, booking, and payment. This domain has 13 constraint slots, that is 5 slots in hotel searching (price, kind, area, stars, hasparking), 5 slots in booking (day, hour, duration, peopleno, surname), and 3 slots in payment sub-domain (address, cardno, surname). Dialogue management over this dialogue domain is cast as a Markov Decision Process (MDP) with the following specification.

- *State*: the belief state $s \in \mathcal{S}$ with 239 dimensions that captures distribution over user's intents and requestable slots
- *Action set* $\mathcal{A}$: 44 dialogue actions, which consists of 8 slot-independent actions and 36 slot-dependent actions.
- *Reward*: -1 at each turn, and 0 or 20 (failed or success dialogue) at the end of dialogue
- *Discount factor* γ: 0.95
- *Maximum number of turns*: 30

4.2 User Simulator

We used an agenda-based user simulator (Schatzmann et al., 2007) with which the belief states perfectly capture the user intent. At the start of each dialogue, the simulated user randomly sets its goal that consists of searching for a hotel and either booking it or paying for it. User will proceed to the booking or payment sub-domain only after achieving the goal of the hotel searching sub-domain.

At the beginning of each sub-domain execution, the user's goal for that sub-domain is randomly generated using database. The agenda is populated by converting all goal constraints into *inform* acts, and all goal requests into *request* acts. For instance, *inform(price=moderate)* indicates a user requirement, and *request(address)* indicates the user asking for the address of the hotel returned by the system. Furthermore, in different dialogue episodes, the simulated user might convey its requirements (i.e. slot values) within a sub-domain to the dialogue system in different orders.

4.3 Dialogue Frameworks

Implementation As the benchmarks without sub-domain, we used flat RL algorithms (i.e. DQN, and PG with REINFORCE). For the benchmark with manual modeling, we used the framework introduced by Budzianowski et al. (2017), which utilized hierarchical Gaussian Process RL (HRL-GP).

All deep (flat and hierarchical) RL agents consist of 2 hidden layers (150 units in layer 1, and 75 (70 for PG) in layer 2). We used Adam optimizer, a mini-batch size of 32, and ϵ-greedy strategy for exploration. In HRL-DQN and HRL-PG_DQN agents, top-level and low-level policies have separate policy networks, each of which has 2 hidden layers as specified above. In these agents, the low-level policies share the same policy network. During execution, we pass the information of the option taken by the top-level policy to the low-level policy network. In HRL-OC and HRL-OC_PVF agents, the policy, the critic Q_Ω, and the termination networks share the same 2 hidden layers, but each of them has its own output layer.

For discovering PVFs, we generated state transition samples using hand-crafted rules (Ultes et al., 2017). We sub-sampled 1,000 unique states using trajectory sampling, and built W from them.

Prior Knowledge In the manual sub-domain modeling, the agent has two types of prior knowledge as follows.

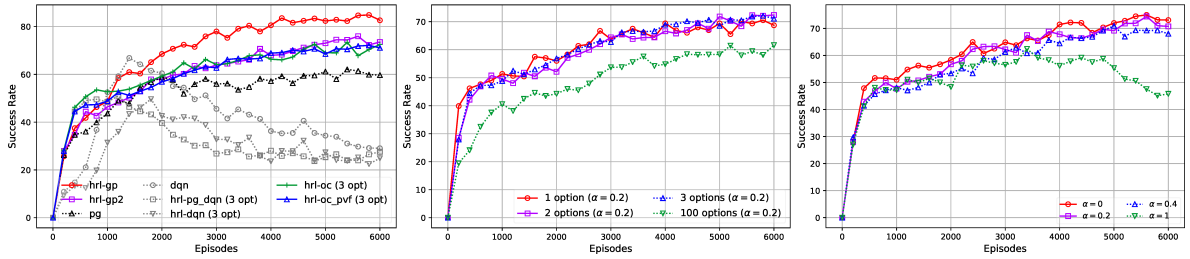
- *sub-domains comprising a dialogue (i.e. hotels, booking, payment).*
- *a valid action set for each sub-domain.* All sub-domains share the same 8 slot-independent actions, but each of them has its own slot-dependent actions.

To assess the impact of each type of prior knowledge, we implemented an HRL-GP framework that uses both types of knowledge and its variant HRL-GP2 that uses only sub-domain information. Both frameworks have separated policies to handle each sub-domain, but HRL-GP2 deals with a more complex situation since it has to select an action from the union of actions sets from all sub-domains, that is 44 dialogue actions in total. Unlike HRL-GP and HRL-GP2, our frameworks with autonomous modeling (HRL-DQN, HRL-PG_DQN, HRL-OC, HRL-OC_PVF) cannot access any prior knowledge. They initially perceive dialogues as a single domain problem and attempt to discover the meaningful sub-domains.

Evaluation We trained each policy in the frameworks for 30 iterations, each of which consists of 200 episodes. In the end of each iteration, we evaluated the performance of the models using 200 episodes. The metric we used for evaluation is the average success rate (SR) of dialogues.

Benchmark	SR(%)	Our framework	SR(%)
FlatRL (DQN)	66.9	HRL-DQN	49.6
FlatRL (PG)	62.0	HRL-PG_DQN	51.0
HRL-GP	84.8	HRL-OC	73.4
HRL-GP2	75.9	HRL-OC_PVF	72.1

Table 1: Highest SR of dialogue agents.



(a) SR of different frameworks (b) HRL-OC_PVF w/ different # of options (c) HRL-OC_PVF with different α

Figure 3: Learning curves of different dialogue frameworks

5 Experimental Results

5.1 Success Rate

The experimental results are shown in Table 1 and Figure 3. First, the flat RL, which is a DQN, achieved an SR of up to 66.9%, but it was unstable. The more stable flat framework, PG, obtained 62%. Our frameworks with autonomous modeling (HRL-OC and HRL-OC_PVF) outperformed flat RL significantly. However, HRL-DQN and HRL-PG_DQN performed worse than flat RL. This suggests that using only intrinsic rewards from PVFs is not adequate for constructing sub-domains that are effective in accumulating extrinsic rewards. Since HRL-OC optimizes its options for maximizing the accumulated extrinsic reward, it has a better SR compared to HRL-DQN and HRL-PG_DQN, which did not use any extrinsic rewards.

The frameworks with manual modeling, i.e. HRL-GP and HRL-GP2, reached an SR of 84.8 and 75.9%, respectively. One of the frameworks with autonomous modeling (i.e. HRL-OC) achieved up to 73.4%. Note that, in HRL-OC, all primitive actions are used for each option, which is the same as HRL-GP2. Although HRL-OC does not have any prior knowledge about sub-domains in a dialogue, it is competitive with the framework with strong supervision on sub-domains. This indicates that HRL-OC is able to learn effective subgoals in a composite-task dialogue.

As shown in Figure 3, learning curves of different dialogue frameworks are examined. Figure 3(a) shows that HRL-OC and HRL-OC_PVF have steeper learning curves than HRL-GP in the first 1000 episodes, which indicates that our frameworks can shorten learning time. Figure 3(b) reports that the use of 2 or 3 options is optimal. Using too many options is harmful because the agent will require more episodes to learn the optimal policy over options. Figure 3(c) shows the effect of the interpolation ratio α for combining both *extrinsic* and *intrinsic* rewards on the SR. However, PVFs seem ineffective with respect to SR. To have $\alpha > 0$ reduces the SR of HRL-OC_PVF with 3 options.

5.2 Discovered Sub-domains

According to our observation, the HRL-OC_PVF with $\alpha = 0.2$, however, discovered more meaningful sub-domains than HRL-OC. To assess the meaningfulness of the discovered sub-domains, we examined how similar these sub-domains to those inherent in the user's agenda. We judge the similarity using the average dialogue turn distance between the turn when the user simulator enters a sub-domain and that when the agent switches sub-domains. The ideal case is to have a turn distance of 1, i.e., once a user enters a sub-domain, the agent responds by switching the active option in the next turn. Table 2 shows that compared to HRL-OC, the integration of PVFs results in sub-domains whose boundaries are similar to those of the user's sub-domains.

Table 3 shows that the integration of PVFs into HRL-OC makes the agent capable of changing the active sub-domain soon after the user enters a sub-domain. This indicates that PVFs can detect interesting belief states. In our further examination, PVFs successfully discover states that indicate dialogue goal, sub-task switching, and request of alternatives from the sampled transitions.

Agents	i^{th} iteration				
	1	2	3	4	5
HRL-OC	2.8	2.8	4.0	3.9	5.3
HRL-OC_PVF ($\alpha = 0.2$)	2.6	2.5	3.0	2.8	2.8
HRL-OC_PVF ($\alpha = 0.4$)	2.9	3.0	3.4	3.2	3.9

Table 2: Average turn distance between activation of user's sub-domains and agent's

User Goal

slots	values	slots	values
hotel searching		payment	
price	don't care	address	bristol
kind	don't care	cardNo	600
area	west	surname	wilson
stars	3		
hasparking	don't care		

requestable slots
name of the hotel
hasinternet
postcode

S: How may I help you?
U: I'm looking for a hotel in west.
S: A hotel in the west area right?
U: Yes, and 3 stars.
S: Would you like a hotel in the cheap, moderate, or expensive price range?
U: any price
S: Alpha-Milton guest house is a nice hotel.
U: Does it have internet connection?
S: It doesn't have internet.
U: post code
S: Their postcode is cb4 1xa.
U: I'll pay with CC by Wilson. [user enters payment]
S: How long is your stay?
U: I don't care, my surname is Wilson.
S: Your name is Wilson right? [system is in payment]
U: Yes, and my address is Bristol.
(...21 dialogue turns to CC alternatives...)
S: I will pay the room with CC 600 by Matt Wilson.
U: Goodbye.

Table 3: Sample dialogue by HRL-OC_PVF (S: agent, U: user). Different font styles and colors indicate different sub-domains activated by agent.

5.3 Discussion

Our experiments show that our proposed framework outperforms the baseline, and is competitive with with the framework with manually defined sub-domains. Even though the experiments are done using a simulator, the simulated user produces dialogue behavior realistic enough for training and testing. As mentioned in Section 4.2, the simulated user specifies its requirements within a sub-domain to the dialogue system in a random order. In addition, the simulator may also not specify several slot values. Such a behavior simulates a situation in which a human user forgets to specify some goal constraints.

In the experiments, the simulator has a constraint, that is it executes the inherent sub-domains in a fixed order. The fixed order of sub-domains, i.e. hotel search and then followed by either booking or payment, can still simulate the real world conversational data, since an activity of reserving a hotel room is commonly accomplished in such order. In other tasks, however, a fixed order of inherent sub-domains may not simulate the real conversation well. Nevertheless, even when the order of the inherent sub-domains are not fixed, we suggest that our proposed framework could still discover options that imitate the inherent sub-domains. This holds when the inherent sub-domains are executed sequentially, and the environment dynamics within each inherent sub-domain is invariant to the execution order of the sub-domains. Another challenging situation is when the inherent sub-domains are executed in an interleaved manner. This simulates a scenario in which a user frequently switches the active sub-domain before the current sub-domain is fulfilled. A further investigation is required to examine the options discovered in such a situation.

6 Conclusion

We proposed a framework that autonomously discovers sub-domains for a composite-task dialogue. Experimental results shows that our framework with autonomous modeling is competitive with the framework with manually defined sub-domains. Analysis also showed that the integration of PVFs leads to meaningful sub-domains.

For future work, we consider the adjustment of the PVFs construction, such as the distance metric between states, the construction of the adjacency matrix, and the use of successor representation (Dayan, 1993; Barreto et al., 2017). We may also need to further examine the discovered options when the inherent sub-domains are executed in several different manners and orders. Finally, it is also interesting to investigate the effectiveness of reusing the learned options in other related dialogue domains.

References

Pierre-Luc Bacon, Jean Harb, and Doina Precup. 2017. The Option-Critic Architecture. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 1726–1734, San Francisco, California, USA.

André Barreto, Will Dabney, Rémi Munos, Jonathan J. Hunt, Tom Schaul, David Silver, and Hado P. van Hasselt. 2017. Successor features for transfer in reinforcement learning. In *Advances in Neural Information Processing Systems 30*, pages 4058–4068, California, USA.

Paweł Budzianowski, Stefan Ultes, Pei-Hao Su, Nikola Mrkšić, Tsung-Hsien Wen, Iñigo Casanueva, Lina

Rojas-Barahona, and Milica Gašić. 2017. Sub-domain Modelling for Dialogue Management with Hierarchical Reinforcement Learning. In *Proceedings of the 18th Annual SIGDIAL Meeting on Discourse and Dialogue*, pages 86–92, Saarbrücken, Germany.

Iñigo Casanueva, Paweł Budzianowski, Pei-Hao Su, Nikola Mrkšić, Tsung-Hsien Wen, Stefan Ultes, Lina Rojas-Barahona, Steve Young, and Milica Gašić. 2017. A Benchmarking Environment for Reinforcement Learning Based Task Oriented Dialogue Management. *CoRR*, abs/1711.1.

Heriberto Cuayáhuitl. 2009. *Hierarchical Reinforcement Learning for Spoken Dialogue Systems*. Ph.D. thesis, University of Edinburgh.

Heriberto Cuayáhuitl, Seunghak Yu, Ashley Williamson, and Jacob Carse. 2016. Deep Reinforcement Learning for Multi-Domain Dialogue Systems. In *Advances in Neural Information Processing Systems 29 Workshop on Deep Reinforcement Learning*, Barcelona, Spain.

Peter Dayan. 1993. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624.

Thomas G. Dietterich. 2000. Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition. *Journal of Artificial Intelligence Research*, 13:227–303.

M. Gasic, N. Mrksic, P. H. Su, D. Vandyke, T. H. Wen, and S. Young. 2016. Policy committee for adaptation in multi-domain spoken dialogue systems. *2015 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2015 - Proceedings*, pages 806–812.

Milica Gašić and Steve Young. 2014. Gaussian processes for POMDP-based dialogue manager optimization. *IEEE Transactions on Audio, Speech and Language Processing*, 22(1):28–40.

Tejas D. Kulkarni, Karthik R. Narasimhan, Ardavan Saeedi, and Joshua B. Tenenbaum. 2016. Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation. In *Advances in Neural Information Processing Systems 29*, pages 3675–3683, Barcelona, Spain.

Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. 2017. End-to-End Task-Completion Neural Dialogue Systems. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP*, pages 733–743, Taipei, Taiwan.

Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. 2017. A Laplacian Framework for Option Discovery in Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, pages 2295–2304, Sydney, NSW, Australia.

Sridhar Mahadevan. 2007. Proto-value Functions : A Laplacian Framework for Learning Representation and Control in Markov Decision Processes. *Journal of Machine Learning Research*, 8:2169–2231.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Ronald Parr and Stuart Russell. 1997. Reinforcement learning with hierarchies of machines. In *Advances in Neural Information Processing Systems 10*, pages 1043–1049, Denver, Colorado, USA.

Baolin Peng, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Celikyilmaz, Sungjin Lee, and Kam-Fai Wong. 2017. Composite Task-Completion Dialogue Policy Learning via Hierarchical Deep Reinforcement Learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017*, pages 2231–2240, Copenhagen, Denmark.

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young. 2007. Agenda-based user simulation for bootstrapping a pomdp dialogue system. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 149–152. Association for Computational Linguistics.

Richard S. Sutton, Doina Precup, and Satinder Singh. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2):181–211.

Stefan Ultes, Lina Rojas-barahona, Pei-Hao Su, David Vandyke, Dongho Kim, Paweł Budzianowski, and Nikola Mrksic. 2017. PyDial : A Multi-domain Statistical Dialogue System Toolkit. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*, pages 73–78, Vancouver, Canada.

Jason D. Williams, Kavosh Asadi, and Geoffrey Zweig. 2017. Hybrid Code Networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*, pages 665–677, Vancouver, Canada.

R. J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D. Williams. 2013. POMDP-based statistical

spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.