To Combine or Not To Combine? A *Rainbow* Deep Reinforcement Learning Agent for Dialog Policy

Dirk Väth **Ngoc Thang Vu**
University of Stuttgart, Institute for Natural Language Processing (IMS)
Pfaffenwaldring 5B, 70569 Stuttgart, Germany
{dirk.vaeth,thang.vu}@ims.uni-stuttgart.de

Abstract

We explore state-of-the-art deep reinforcement learning methods such as prioritized experience replay, double deep Q-Networks, dueling network architectures, distributional learning methods for dialog policy. Our main findings show that each individual method improves the rewards and the task success rate but combining these methods in a *Rainbow* agent, which performs best across tasks and environments, is a non-trivial task. We, therefore, provide insights about the influence of each method on the combination and how to combine them to form the *Rainbow* agent.

1 Introduction

Dialog system can be designed for generic purposes, e.g. smalltalk (Weizenbaum, 1966) or a specific task such as finding restaurants or booking flights (Bobrow et al., 1977; Wen et al., 2017). This paper focuses on task-oriented dialog systems, which interact with a user to aid achieving their goals. The systems have several modules which solve different subtasks (Williams et al., 2016) starting with natural language understanding (NLU) module (De Mori et al., 2008). Its output is then passed to a belief tracking module (Mrkšić et al., 2017) that holds the state of the dialog, i.e. all relevant information provided by the user. This belief state is then passed to the dialog policy module (Williams and Young, 2007) which has to decide how the system should reply. Depending on the ontology of the task, e.g. the restaurant search, the size of the input space for the policy can quickly become very large. Furthermore, the belief state might be wrong due to noisy inputs, e.g. the user could be misunderstood because of NLU errors or in general, language ambiguity. Therefore, building such policies by hand is rather time consuming. Reinforcement learning (RL) can alleviate this task by allowing to learn such policies automatically (Williams and Young, 2007) with a user simulator such as proposed in Schatzmann et al. (2007) within a task (Dhingra et al., 2017; Peng et al., 2018), between task and non-task (Yu et al., 2017) and also in multimodal dialog systems (Manuvinakurike et al., 2017; Zhang et al., 2018).

Deep RL has been proven to be successful with Deep Q-Learning (DQN) (Mnih et al., 2013) introducing the idea of using neural networks as a Q-function approximator. It has been widely used in the context of dialog policy learning (Fatemi et al., 2016; Dhingra et al., 2017; Casanueva et al., 2017). However according to a recent comparison (Casanueva et al., 2017) in the context of dialog policy learning, it performed worse than other RL methods such as Gaussian Process in many testing conditions. Recently, several advances in deep RL such as distributional RL (Bellemare et al., 2017) , dueling network architectures (Wang et al., 2016) and their combination (Hessel et al., 2018) - a *Rainbow* agent - have been shown to be promising for further improvements of deep RL agents in benchmark environments, e.g. Atari 2600. However, it is still unclear whether these methods could advance dialog policies.

This paper attempts to provide insights motivated from dialog policy modeling perspectives how to use state-of-the-art deep RL methods such as prioritized experience replay (Schaul et al., 2015), double DQN (Van Hasselt et al., 2016), dueling network architecture, distributional learning method and how to combine them to train the *Rainbow* agent for dialog policy learning [1]. Moreover, we explore the influence of each method w.r.t the resulting rewards and the number of successful dialogs, highlighting methods with the biggest and the smallest impact.

[1] Agent code: https://github.com/DigitalPhonetics/adviser

Proceedings of the SIGDial 2019 Conference, pages 62–67
Stockholm, Sweden, 11-13 September 2019.

	Env. 1			Env. 2			Env. 3			Env. 4			Env. 5			Env. 6		
Task	T1.1	T1.2	T1.3	T2.1	T2.2	T2.3	T3.1	T3.2	T3.3	T4.1	T4.2	T4.3	T5.1	T5.2	T5.3	T6.1	T6.2	T6.3
Domain	CR	SFR	LAP	CR	SFR	LAP	CR	SFR	LAP	CR	SFR	LAP	CR	SFR	LAP	CR	SFR	LAP
SER	0%			0%			15%			15%			15%			30%		
Masks	On			Off			On			Off			On			On		
User	Standard			Standard			Standard			Standard			Unfriendly			Standard		

Table 1: Benchmarking environments with several domains, semantic error rates (SERs), action masks and user models (Casanueva et al., 2017).

2 Proposed Method

For value-based reinforcement learning methods like Q-learning, potentially large state spaces as in the dialog setting require the use of function approximators. The DQN-Algorithm (Mnih et al., 2013) is an example of such a method where the action-value function is approximated by a neural network which takes a state vector as input and outputs a value for each possible action. Loss is calculated with the squared temporal difference (TD) error. Efficient off-policy batch-training is enabled by a replay buffer which records the agent's turn-level experiences and allows the drawing of uncorrelated training samples.

Prioritized experience replay Drawing samples from this buffer uniformly is straightforward but problematic: important state transitions might never be drawn from the buffer or at least too few times to have an impact on the network weights. Motivated by the insight that a high absolute TD-error of an experience means that the current action-value is not an accurate estimate yet, prioritized experience replay (Schaul et al., 2015) samples experiences having higher TD-errors with greater probability than those with lower TD-error. This method is relevant because it is expected to increase learning efficiency. In the context of dialog policy, there are some system actions which are crucial to the outcome of the dialog and should have a higher probability for being used as training data if they are not well approximated. For example, if systems end the dialog before the user's goal is completed by telling the user *goodbye*, this will immediately terminate a dialog with a negative reward and without any chance of recovery.

Double DQN Another improvement mitigates the overestimation bias inherent to Q-learning by introducing a second action-value network (Van Hasselt et al., 2016) which copies the parameters from the online action-value network periodically and is held fix otherwise. This additional network is then used to evaluate the action-value of the action selected greedily w.r.t. the online Q-function, thereby decoupling action choice and evaluation which could increase stability of the learning process.

Dueling network architecture In comparison to the action-value function, the state-value function is a simpler estimate - it is the expectation over a state's action-values and therefore only a single value. But in states where the action choice does not matter, or to avoid visiting states with a low state value in general, an estimate of the value function should be sufficient. Dueling network architecture (Wang et al., 2016) therefore splits the calculation of the action-value function into separate layers of a neural network, one group computing the value function and another an advantage function chosen so that their combination results in the action-value function again. This approach also has the benefit that the state value estimation is updated every time when a state is observed by the network, regardless of the chosen action. As a result, it should encourage generalization across actions. In dialog settings, there are many states where generalization across actions could prove beneficial, e.g. exact action choice is not important, just the choice between action classes. For example at the beginning of a dialog, when users greet the system without providing any information, the only appropriate action for the system is to ask for more information. The exact type of information should not matter and all other actions except for the dialog ending action should be about equally unsuitable.

Distributional learning method One of the latest additions to reinforcement learning is the quantile regression distributional reinforcement learning algorithm (Dabney et al., 2018). Instead of learning only the expected value for each state-action pair, as in regular Q-learning, the distribution of rewards is approximated instead, thereby modeling the randomness of the reward over multiple turns induced by action selection and random state transitions. A noisy environment like dialog could benefit from better knowledge about the distribution of rewards.

The Rainbow agent Following the methodology from (Hessel et al., 2018), we extend the DQN algorithm (Mnih et al., 2013) with prioritized experience replay, double DQN, and dueling network architecture. Furthermore in contrast to (Hessel et al., 2018), we apply the following changes to successfully train the *Rainbow* agent: 1) we drop the multi-step method (Sutton, 1988) because it seems to diminish the obtained rewards. As the step size gets larger, the rewards are decreased more. A possible explanation could be that the noise generated by the user simulator leads to accumulation of noise in rewards over multiple steps, which could lead to higher variance in value estimates. 2) we discard the noisy linear layers (Fortunato et al., 2018), relying on ϵ-greedy exploration instead. The first reason could be the additional parameters, which usually would require more training samples. Since the agent was already required to learn environmental noises from the user simulator, a complementary explanation could be that the inclusion of a second noise distribution might have been too difficult to learn, especially when considering the relatively small amount of training episodes. 3) we swap the categorical DQN approach (Bellemare et al., 2017) with the quantile regression Q-learning algorithm (Dabney et al., 2018), now consistent with the theoretic results from (Bellemare et al., 2017), no longer restricting the values of the value distribution to a uniform resolution and also no longer requiring knowledge about their bounds.

3 Resources

We used PyDial toolkit (Ultes et al., 2017) as a test-bed for experiments and evaluation. It includes a configurable user simulator and provides multiple dialog ontologies like Cambridge Restaurants (CR), Laptops (LAP) and San Francisco Restaurants (SFR). The ontologies used for the benchmarks in this paper together with their properties are listed in table 2.

Domain	#slots	#requests	#values
CR	3	9	268
SFR	6	11	636
LAP	11	21	257

Table 2: Benchmark domains with #slots the user can provide or #request from the system as well as #values of each requestable slot (Casanueva et al., 2017).

Casanueva et al. (2017) propose six different environmental models, varying in user friendliness, simulated input channel noise and the presence or absence of action masks, which, when enabled, simplify learning by masking some of the possible actions. An overview of all these environmental configurations and their assignment to tasks is given in Table 1. Evaluation results in Casanueva et al. (2017) with several dialog policy types, e.g. a handcrafted policy and the best reported policies serve as baselines in our experiments.

4 Experimental Results

Training and evaluation with the PyDial user simulator follows the PyDial benchmarking tasks (Casanueva et al., 2017), where each task (see Table 1) is trained on 10000 dialogs split into ten training iterations of 1000 dialogs each. We evaluate policies after each training iteration on 1000 test dialogs. All of the following results were obtained by averaging over the outcome of ten different random seeds using the parameters described in appendix A.

4.1 The *Rainbow* Agent

The first row of Table 3 and 4 show the results of the highest scoring policy from the PyDial benchmark (Casanueva et al., 2017) to serve as baselines. Evaluations of the handcrafted policies follow in the last line. The results show that *Rainbow* agent outperforms reward of the best PyDial agents in all 18 conditions and success rate in 16 out of 18 setting. Compared to the basic DQN agent, *Rainbow* agent is better in all 18 conditions w.r.t both reward and success rate. When averaged across all 18 tasks, *Rainbow* agent (mean reward 10.1) scores more than 29% higher rewards compared to the best PyDial agent (DQN, mean reward 7.8) and more than 9.7% compared to our DQN agent. An average success rate of 90.4% is superior to the best PyDial agent (GP-Sarsa, 80.2%). Mean deviation across all tasks and random seeds is 0.4 in reward and 1.6% in successful dialogs.

4.2 Model Ablation Analysis

Figure 1 shows the averaged success rates for each of our *Rainbow* agents leaving out one particular method after training with 10000 dialogs. Each of the plotted values has been evaluated on 1000 dialogs per random seed and averaged over all tasks. Regarding learning speed w.r.t. success rate, the

Agent	Task																	
	T1.1	T1.2	T1.3	T2.1	T2.2	T2.3	T3.1	T3.2	T3.3	T4.1	T4.2	T4.3	T5.1	T5.2	T5.3	T6.1	T6.2	T6.3
best PyDial	13.5[1]	12.3[2]	11.0[2]	12.7[3]	10.1[1]	9.1[3]	12.2[3]	8.6[2]	6.5[2]	11.1[3]	8.2[3]	5.8[1]	10.5[3]	6.5[2]	3.8[2]	9.9[3]	3.6[3]	3.2[2]
DQN	13.0	10.8	9.5	13.1	11.0	9.5	12.7	9.7	7.5	11.9	7.9	5.1	11.4	7.3	4.3	10.7	5.7	4.7
Rainbow	**14.0**	**12.4**	**11.2**	**13.6**	**11.8**	**10.1**	**12.8**	**9.8**	**8.1**	**12.2**	**10.0**	**8.9**	**11.8**	**7.8**	**4.9**	**10.9**	**6.5**	**4.8**
handcrafted	14.0	12.4	11.7	14.0	12.4	11.7	11.0	9.0	8.7	11.0	9.0	8.7	9.7	6.4	5.5	9.3	6.0	5.3

Table 3: Rewards per task and agent ([1]GP-SARSA, [2]eNAC, [3]DQN).

Agent	Task																	
	T1.1	T1.2	T1.3	T2.1	T2.2	T2.3	T3.1	T3.2	T3.3	T4.1	T4.2	T4.3	T5.1	T5.2	T5.3	T6.1	T6.2	T6.3
best PyDial	99.4[1]	**97.3**[1]	92.1[2]	97.9[1]	**95.4**[1]	87.5[1]	95.8[1]	84.1[2]	73.3[2]	92.6[1]	81.1[3]	74.0[1]	92.6[3]	82.3[2]	72.8[2]	89.6[1]	64.8[3]	61.2[2]
DQN	95.1	89.4	83.7	96.9	91.4	87.6	97.1	89.6	79.6	94.3	79.8	68.0	95.6	84.9	74.4	91.7	75.8	71.1
Rainbow	**99.7**	**97.3**	**93.4**	**98.8**	94.4	**90.3**	**97.2**	**90.5**	**83.6**	**96.5**	**88.8**	**87.3**	**97.0**	**87.9**	**78.0**	**92.4**	**80.4**	**73.0**
handcrafted	100.0	98.2	97.0	100.0	98.2	97.0	96.7	90.9	89.6	96.7	90.9	89.6	95.9	87.7	85.1	89.6	79.0	76.1

Table 4: Success rates per task and agent ([1]GP-SARSA, [2]eNAC, [3]DQN).

results show that *Rainbow* agent without distributional learning method learns the fastest, surpassing the final success rate of the DQN and non-dueling agents after only 2000 dialogs. The reward plot displays similar characteristics.

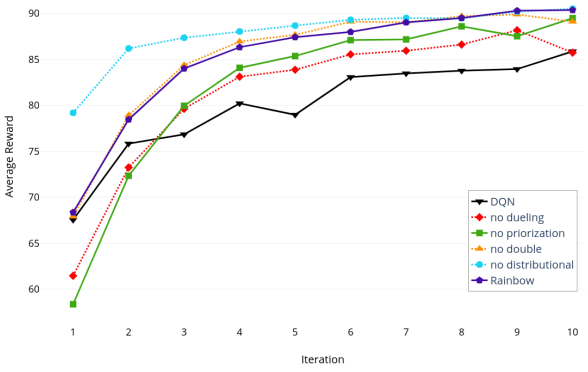

Figure 1: Avg. success rate for policies after training with 10000 dialogs (1000 dialogs per iteration).

Results in Table 5 show that there is almost no difference between the distributional and the non-distributional approach. Their final rewards are the same and their success rates differ by 0.1% when averaged across all tasks. A possible reason could be that the diversity of the training dialogs was too little and rewards too sparse to show a benefit by using the distributional reinforcement learning method. This coincides with the findings in Hessel et al. (2018) which found their combined agent without distributional learning performing similar to the combined agent with distributional learning for the first 40 million frames on the Atari benchmark.

The strongest benefits to final performance come with the dueling architecture. For some scenarios like the previously described dialog start without any user-provided information, we examined the action-state values by clustering them and observed fewer clusters and smaller within-cluster variance for the dueling agents, indicating better generalization and simpler action-value functions. Prioritized experience replay helped with learning efficiency but had no significant effect on final performance, as expected. Only a small improvement can be attributed to double DQN, but overall performance seems to be slightly more stable.

Overall, Table 5 shows that the final best *Rainbow* agent performs considerably better than the best reported PyDial agent and the DQN agent across all the tasks and testing environments and is on par with handcrafted policy performance.

Agent	CR		SFR		LAP	
	Suc.	Rew.	Suc.	Rew.	Suc.	Rew.
best PyDial	94.5%[1]	10.7[1]	79.7%[1]	6.8[2]	66.8%[2]	5.0 [2]
DQN	95.1%	12.2	85.1%	8.7	77.4 %	6.8
Rainbow	96.9%	12.6	89.9%	9.7	84.3%	8.0
- distributional	96.6%	12.4	89.6%	9.6	85.4%	8.2
- double	96.0%	12.3	88.5%	9.4	85.2%	8.3
- dueling	95.5%	12.1	84.7%	8.4	77.0%	6.4
- prioritization	97.4%	12.7	89.0%	9.7	82.1%	8.0
handcrafted	90.8%	9.2	90.8%	9.2	89.1%	8.6

Table 5: Success rates and rewards per domain ([1]GP-SARSA, [2]DQN).

5 Conclusions

We explored state-of-the-art deep RL methods for dialog policy on different domains with various noise levels and user behaviours. Our findings are that not all extensions to DQN prove beneficial in dialog policy settings, especially when learning speed is concerned: distributional reinforcement learning method requires more training time to reach the success rates and final rewards of the non-distributional agent. The *Rainbow* agent that makes use of prioritized experience replay, double DQN and dueling network architecture is stable across domains and evaluation settings and learns fastest (when excluding distributional method).

References

Stefan Ultes et al. 2017. PyDial: A Multi-domain Statistical Dialogue System Toolkit. In *ACL, System Demonstrations*.

Marc G Bellemare, Will Dabney, and Rémi Munos. 2017. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*.

Daniel G. Bobrow, Ronald M. Kaplan, Martin Kay, Donald A. Norman, Henry Thompson, and Terry Winograd. 1977. GUS: a Frame-Driven Dialog System. *Artificial Intelligence*, 8.

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. In *Deep Reinforcement Learning Symposium*.

Will Dabney, Mark Rowland, Marc G Bellemare, and Rémi Munos. 2018. Distributional reinforcement learning with quantile regression. In *AAAI*.

Renato De Mori, Frédéric Bechet, Dilek Hakkani-Tur, Michael McTear, Giuseppe Riccardi, and Gokhan Tur. 2008. Spoken language understanding. *IEEE Signal Processing Magazine*, 25(3).

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards end-to-end reinforcement learning of dialogue agents for information access. In *ACL (Vol. 1: Long Papers)*.

Mehdi Fatemi, Layla El Asri, Hannes Schulz, Jing He, and Kaheer Suleman. 2016. Policy networks with two-stage training for dialogue systems. In *SigDial*.

Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, et al. 2018. Noisy networks for exploration. In *International Conference on Learning Representations*.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. 2018. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI*.

Ramesh Manuvinakurike, David DeVault, and Kallirroi Georgila. 2017. Using reinforcement learning to model incrementality in a fast-paced dialogue game. In *SigDial*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*.

Nikola Mrkšić, Diarmuid O Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *ACL (Vol. 1: Long Papers)*.

Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, Yun-Nung Chen, and Kam-Fai Wong. 2018. Adversarial advantage actor-critic model for task-completion dialogue policy learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing*.

Jost Schatzmann, Blaise Thomson, Karl Weilhammer, Hui Ye, and Steve Young. 2007. Agenda-based user simulation for bootstrapping a pomdp dialogue system. In *NAACL (Vol. 2: Short Papers)*.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2015. Prioritized experience replay. In *International Conference on Learning Representations*.

Richard S Sutton. 1988. Learning to predict by the methods of temporal differences. *Machine learning*.

Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep reinforcement learning with double q-learning. In *AAAI*.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. 2016. Dueling network architectures for deep reinforcement learning. In *International Conference on Machine Learning*.

Joseph Weizenbaum. 1966. ELIZA: A Computer Program for the Study of Natural Language Communication Between Man and Machine. *Communications of the ACM*, 9(1).

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *EACL*.

Jason Williams, Antoine Raux, and Matthew Henderson. 2016. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33.

Jason D Williams and Steve Young. 2007. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422.

Zhou Yu, Alan W Black, and Alexander I Rudnicky. 2017. Learning conversational systems that interleave task and non-task content. In *International Joint Conference on Artificial Intelligence*.

Jiaping Zhang, Tiancheng Zhao, and Zhou Yu. 2018. Multimodal hierarchical reinforcement learning policy for task-oriented visual dialog. In *SigDial*.

A Hyperparameters

All neural network layers are fully connected linear layers with ReLUs as activation functions. In case of the dueling network architecture, the shared layer consists of 256 neurons, followed by two value layers, each with 300 neurons, and two advantage layers with 400 neurons per layer. Distributional agents use an atom count of 50. Where the dueling architecture is replaced by a standard architecture in the evaluation process, three layers of sizes 256, 700 and 700 are used to guarantee a fair comparison to the dueling case by providing the same model capacity. For prioritized replay, the prioritization exponent α is set to 0.525 and importance sampling exponent β to 0.4 (Schaul et al., 2015). To train the networks we use the Adam optimizer with a learning rate of 10^{-4}. Exploration is performed ϵ-greedy with linear ϵ decay, starting at 0.3. Whenever an agent makes use of double Q-learning, it updates its target network after 6 dialogs. All agents use an experience replay buffer capacity of 16384 transitions, a discount factor $\gamma = 0.99$, minibatch size 256 and the Huber Loss κ is set to 1. Following the PyDial benchmarking process, we leave all hyperparameters constant across all environments and dialog domains (Casanueva et al., 2017), thus also evaluating the generalization capabilities of the agents.

B Example Dialogs

Tables 6 and 7 show example dialogs between the *Rainbow* agent as well as the handcrafted policy and a user recorded in the LAP domain as qualitative evaluation. The *Rainbow* agent was trained in a noisy environment with 15% semantic error rate. The user's task was to find a cheap, light notebook with medium-sized hard drive and arbitrary battery rating. User responses were kept identical for all tested policies whenever possible. Both the combined and the handcrafted agents succeeded in fulfilling the user's goal with the same amount of dialog turns.

Turn	Actor	Utterance
1	System	Hello, welcome to the laptop information system, what kind of laptop are you after?
1	User	Hi, I'm looking for a cheap and light laptop
2	System	Would you like the hard drive to be small, medium, or large?
2	User	Medium should suffice
3	System	Sorry would you like a product for business computing or not?
3	User	Yes, please
4	System	The satellite-pro-nb10-a-10p is in the budget price range, is in the light weight range and is in the medium drive range.
4	User	Thank you, bye
5	System	Thank you, goodbye.

Table 6: Recorded dialog between a human and the *Rainbow* agent for task T4.3.

Turn	Actor	Utterance
1	System	Hello, welcome to the laptop information system, what kind of laptop are you after?
1	User	Hi, I'm looking for a cheap and light laptop
2	System	Would you like your battery rating to be standard, good, or exceptional?
2	User	I don't care
3	System	Would you like the hard drive to be small, medium, or large?
3	User	Medium should suffice
4	System	The satellite-pro-nb10-a-10p is in the light weight range, has a standard battery rating , is in the budget price range and is in the medium drive range .
4	User	Thank you, bye
5	System	Thank you, goodbye.

Table 7: Recorded dialog between a human and PyDials handcrafted policy for task T4.3.