A Collaborative Multi-agent Reinforcement Learning Framework for Dialog Action Decomposition

Huimin Wang and **Kam-Fai Wong**
The Chinese University of Hong Kong
wanghm520@gmail.com
kfwong@se.cuhk.edu.hk

Abstract

Most reinforcement learning methods for dialog policy learning train a centralized agent that selects a predefined joint action concatenating domain name, intent type, and slot name. The centralized dialog agent suffers from a great many user-agent interaction requirements due to the large action space. Besides, designing the concatenated actions is laborious to engineers and maybe struggled with edge cases. To solve these problems, we model the dialog policy learning problem with a novel multi-agent framework, in which each part of the action is led by a different agent. The framework reduces labor costs for action templates and decreases the size of the action space for each agent. Furthermore, we relieve the non-stationary problem caused by the changing dynamics of the environment as evolving of agents' policies by introducing a joint optimization process that makes agents can exchange their policy information. Concurrently, an independent experience replay buffer mechanism is integrated to reduce the dependence between gradients of samples to improve training efficiency. The effectiveness of the proposed framework is demonstrated in a multi-domain environment with both user simulator evaluation and human evaluation.

1 Introduction

Dialog policy optimization is one of the most critical tasks of task-oriented dialog modeling. Recently, it has shown great potentials for using reinforcement learning (RL) based methods to formulate dialog policy learning (Li et al., 2017; Peng et al., 2017). However, most of these methods learn a centralized agent based on the joint action space that covers predefined atomic action (Budzianowski et al., 2018), which is the concatenation of domain name, intent type, and slot name, e.g. 'restaurant-inform-address', or both atomic actions and the top-k most frequent atomic action combinations(Lee et al., 2019a). The elaborate concatenated actions may achieve acceptable performance in simple cases, however, continuously suffer from being laborious to engineers and struggled with edge cases in multi-domain or complex scenes. Another drawback of the centralized agent is its exponential growth in the observation and actions spaces with the growing number of domains (Lee et al., 2019b).

To alleviate the problem of large user-agent interaction requirements caused by the large action space, a hierarchical reinforcement learning framework was proposed to learn the dialog policy that operates at different temporal scales (Peng et al., 2017). It has achieved promising results, however, is still up against some challenges. Firstly, their setting requires a rule-based critic to provide the intrinsic reward for the low-level agent. However, creating such a critic is not easy, especially in intricate scenarios. The man-made critic, somewhat inadvertently, may bias the convergent optimal. Moreover, the action space composed of intent and slot for the low-level agent can be still large, especially when there are a lot of intent types and slot names. Drawing the structural features of dialog actions, we address the above problems with a proposed collaborative multi-agent reinforcement learning framework, where the concatenated dialog action space is decomposed into subspaces corresponding to the domain, intent type, and slot name. Furthermore, each subspace is assigned to different agents, which cooperate to make the final joint action without any human knowledge. The agents concatenate together and pass the output to the next agent. To relieve the non-stationary problem (Claus and Boutilier, 1998; Hu and Wellman, 2003) caused by unexpected changes in the dynamics of the environment as evolving of the agents' policies and to reduce the dependence of the gradients due to the non-independent data, we propose a new approach which allows **J**oint **O**ptimization based on **I**ndependent **E**xperience replay buffers for all

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 7882–7889

agents, termed as **JOIE**. Our experiments show that such a multi-agent framework reduces the state-action space size significantly and make exploration more efficient. Furthermore, JOIE leads to a better performance benefit from the proposed optimization mechanism.

To the best of our knowledge, this is the first work that strives to develop a multi-agent RL-based dialog action decomposition framework. Our main contributions are three-fold:

- We formulate dialog policy learning in the mathematical framework of collaborative multi-agent reinforcement learning.
- We propose an efficient and effective multi-agent-based approach factoring the action space size and learning each part by different agents with joint optimization and independent experience replay.
- We validate the effectiveness of the proposed method in a multi-domain task with both user simulators and human users.

2 Related Work

Many studies have been dedicated to optimizing dialog policy with reinforcement learning, most of which learn a centralized agent that maps the observation to a joint action (Young et al., 2013; Su et al., 2016; Williams et al., 2017; Peng et al., 2018a,b; Lipton et al., 2018; Li et al., 2020a; Zhu et al., 2020; Li et al., 2020b; Wang et al., 2020). For more efficient exploration, (Peng et al., 2017) factor the centralized spaces into hierarchical reinforcement learning paradigms.

Meanwhile, cooperative multi-agent reinforcement learning methods have started moving from tabular methods to deep learning methods and are widely applied especially on computer games (Sunehag et al., 2017; Rashid et al., 2018; Jhunjhunwala et al., 2020). Towards multi-agent task-oriented dialog policy, a lot of progress is being made in modeling the interaction as a stochastic collaborative game, where dialog agent and the user simulator are jointly optimized with their objectives (Liu and Lane, 2017; Papangelis et al., 2019; Takanobu et al., 2020). Building a user simulator in this way is more flexible. However, different from existing frameworks, our multi-agent framework is devoted to decompose concatenated actions in order to reduce the large action space size to improve the performance of dialog agents.

3 Approach

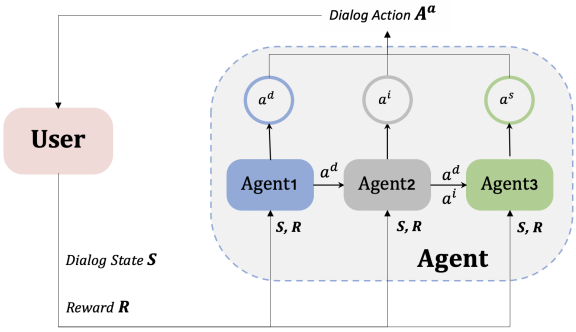


Figure 1: Illustration of the collaborative multi-agent framework for dialog policy learning.

Different from the previous methods that learn a centralized agent or that adopt hierarchical RL paradigms, we cast the policy learning as a multi-agent RL framework, as shown in Figure 1. It integrates three agents specified to be responsible for the domain a^d, intent type a^i, and slot name a^s, respectively. They share reward r and make decisions cooperatively based on the state s from the user. Consequently, a concatenation A_a from the three agents is passed to the user.

3.1 Multi-agent Dialog Policy

Specifically, Agent1 perceives the state s and learns the domain policy π_d that selects a domain category $a^d \in A_d$. Meanwhile, Agent2 equipped with the intent policy π_i, takes as input the state s and the selected domain a^d, and decides the intent type $a^i \in A_i$. Then, Agent3 receives s, a^d and a^i, and determines the slot names $a^s \in A_s$ based on the slot policy π_s. Where A_d, A_i, and A_s are the sets of all possible domain names, intent types, and slot names, respectively.

Naturally, we aim to simultaneously optimize all policies that achieve the maximal shared cumulative rewards. Specifically, Agent1 aims to learn the domain policy π_d that maximizes the expected sum of rewards condition on s and a^d that $\mathbf{E}_{\pi_d, s_t=s, a_t^d=\pi_d(s_t)}\left[\sum_t \gamma^t r_t\right]$, where r_t denotes the reward from the user at turn t, and $\gamma \in [0, 1]$ is a discount factor. Similarly, the intent policy π_i is trained to maximize $\mathbf{E}_{\pi_i, s_t=s, a_t^d=a^d, a_t^i=\pi_i(s_t||a_t^d)}\left[\sum_t \gamma^t r_t\right]$, while Agent3 tries to optimize π_s that maximizes $\mathbf{E}_{\pi_s, s_t=s, a_t^d=a^d, a_t^i=a^i, a_t^s=\pi_s(s_t||a_t^d||a_t^i)}\left[\sum_t \gamma^t r_t\right]$.

All policies can be learned with DQN (Mnih et al., 2015). Concretely, the domain policy estimates the optimal Q-function represented by a

neural network parameterized by θ_d that satisfies the following:

$$Q_{\theta_d}(s, a^d) = \mathbf{E}_{\pi_d}[\, r_t + \gamma \max_{a^d_{t+1}} Q_{\theta'_d}(s_{t+1}, a^d_{t+1}) | s_t = s, a^d_t = a^d \,] \quad (1)$$

Where $Q_{\theta'_d}(.)$ is the target state-action value function that is only periodically updated. Similarly, the intent policy π_i estimates the optimal Q-function parameterized by θ_i that satisfies the following:

$$Q_{\theta_i}(s||a^d, a^i) = \mathbf{E}_{\pi_i}[\, r_t + \gamma \max_{a^i_{t+1}} Q_{\theta'_i}(s_{t+1}||a^d_{t+1}, a^i_{t+1}) | s_t = s, a^d_t = a^d, a^i_t = a^i \,] \quad (2)$$

Where $Q_{\theta'_i}(.)$ is the target value function, and $||$ is the tagger of concatenation. Meanwhile, the slot policy estimates the optimal Q-function parameterized by θ_s that satisfies the following:

$$Q_{\theta_s}(s||a^d||a^i, a^s) = \mathbf{E}_{\pi_s}[\, r_t + \gamma \max_{a^s_{t+1}} Q_{\theta'_s}(s_{t+1}||a^d_{t+1}||a^i_{t+1}, a^s_{t+1}) | s_t = s, a^d_t = a^d, a^i_t = a^i, a^s_t = a^s \,] \quad (3)$$

3.2 JOIE for Policy Learning

To alleviates the dependence of the gradients caused by the non-independent data, the agents maintain their independent experience replay buffer, set as D_d, D_i and D_s for the domain policy, the intent policy, and the slot policy respectively. Consequently, the Q-function Q_{θ_d} for the domain policy is learned by minimizing the following loss function:

$$\mathcal{L}(\theta_d) = \mathbf{E}_{(s,a^d,r,s')\sim D_d}\left[(y^d_i - Q_{\theta_d}(s, a^d))^2\right]$$
$$y^d_i = r + \gamma \max_{(a^d)'} Q_{\theta'_d}(s', (a^d)') \quad (4)$$

Similarly, the intent policy tries to minimize the following loss function:

$$\mathcal{L}(\theta_i) = \mathbf{E}_{(s,a^d,a^i,r,s',(a^d)')\sim D_i}[\, (y^i_i - Q_{\theta_i}(s||a^d, a^i))^2 \,]$$
$$y^i_i = r + \gamma \max_{(a^i)'} Q_{\theta'_i}(s'||(a^d)', (a^i)') \quad (5)$$

Meanwhile, the loss function for the slot policy is:

$$\mathcal{L}(\theta_s) = \mathbf{E}_{(s,a^d,a^i,a^s,r,s',(a^d)',(a^i)')\sim D_s}[\, (y^s_i - Q_{\theta_s}(s||a^d||a^i, a^s))^2 \,]$$
$$y^s_i = r + \gamma \max_{(a^s)'} Q_{\theta'_s}(s'||(a^d)'||(a^i)', (a^s)') \quad (6)$$

As shown in Figure 1 and Equation 4, 5 and 6, all agents can observe the global state and the previous agents' actions during training. This setting stabilizes the training procedure by alleviating the non-stationary environment caused by unexpected changes in the dynamics as evolving of the agents' policies. Besides, we proposed to utilize a joint optimization process by adding up each agent's losses represented as Equation 7 based on a shared hidden network. With the joint optimization, the agents do not experience unexpected changes in the environment because different agents can exchange policy information through the shared hidden layers ϕ.

$$\mathcal{L}(\theta_{d,i,s}; \phi) = \sum_{k\in\{d,i,s\}} \mathcal{L}(\theta_k; \phi) \quad (7)$$

A detailed summary of the learning algorithm of the collaborative multi-agent reinforcement learning for dialog policy based on joint optimization and independent experience replay buffer (JOIE) is provided in Algorithm 1 in Appendix D.

4 Experiments

Comparison is on MultiWoz (Budzianowski et al., 2018) with a public available agenda-based user simulator (Zhu et al., 2020). The detail of the user simulator and implementation is in Appendix B, C. We first evaluate 2-agent based models that factor the centralized spaces into two subspaces of the domain and joint intent-slot on 3 different domains sizes of 2, 4, and 7 on MultiWoz. Then we compare 3-agent based models that decompose the action spaces into three subspaces of the domain, intent, and slot. The dataset contains 7 domains, 13 intents, and 28 slots totally. Details of the dataset are provided in Appendix A.

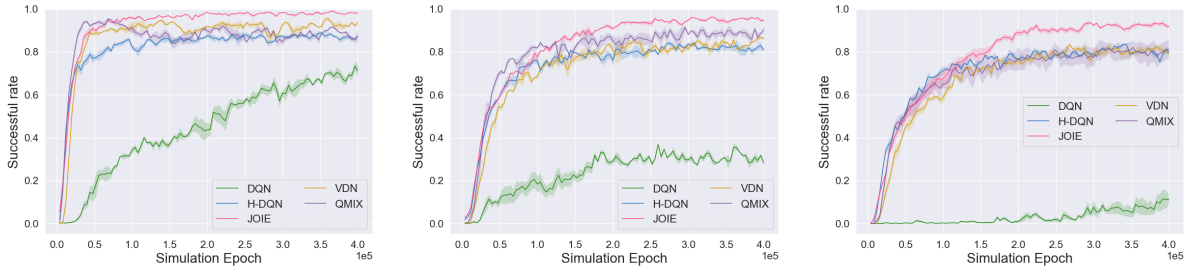
4.1 Baseline Agents

We compare JOIE with DQN, Hierarchical DQN (H-DQN), and two multi-agent RL agents. Note that, we do not consider any other methods that use demonstrations because our motivation is to improve learning in a large action space without human knowledge.

- **DQN**(Mnih et al., 2015) agent is learned with one Deep Q-Network.
- **H-DQN**(Peng et al., 2017) is a hierarchical deep RL approach consists of: (1) a top-level agent that selects domain (sub-goal), (2) a low-level agent that determines intent-slot to complete the sub-goal.

Table 1: The performance of the average turn (Turn) and the average reward (Reward) of the agents in different numbers of domains (termed as #domains). Succ. denotes success rate.

Agent	#domains = 2			#domains = 4			#domains = 7		
	Succ.↑	Turn↓	Reward↑	Succ.↑	Turn↓	Reward↑	Succ.↑	Turn↓	Reward↑
DQN	0.71	13.20	-2.68	0.28	16.29	-27.26	0.11	20.00	-54.90
H-DQN	0.87	7.68	53.08	0.80	9.31	39.76	0.80	10.16	35.95
JOIE	**0.98**	**5.82**	**66.71**	**0.94**	**8.45**	**50.59**	**0.91**	**9.45**	**40.82**
VDN	0.93	8.13	56.05	0.86	10.55	34.00	0.79	10.85	25.10
QMIX	0.87	9.92	49.52	0.90	10.18	42.57	0.81	10.97	29.68



(a) 2 domains. (b) 4 domains (c) 7 domains

Figure 2: Learning curves of the 2-agent based dialog agents trained on different numbers of domains.

- **JOIE** is our proposed collaborative multi-agent framework factoring the joint action space and learning each part by a different agent with joint optimization and independent experience replay, as described in Section 3.2.
- **VDN**(Sunehag et al., 2017) is a multi-agent method that combines each agent's state action-value function as a simple sum for optimization with shared transitions.
- **QMIX**(Rashid et al., 2018) is a variant of VDN which contains a mixing network that centralizes each agent's state action-value function for optimization.

4.2 Main Results

All agents are evaluated with the success rate (Succ.) at the end of the training, average turn (Turn), average reward (Reward). The main simulation results are shown in Table 1 and Figure 2, 3. The results show that the proposed JOIE learns much faster and performs consistently better in cases with a statistically significant margin.

Results of 2-agent based Models Figure 2 shows the learning curves of 2-agent based models. Firstly, JOIE achieves the best Succ. (on average 0.98) with the highest learning efficiency for all domain sizes. Qmix and VDN adopt an optimization fashion that estimates a concatenated action values, which is originally for partial observability. JOIE abandons this step to avoid the extra cost since we assume the state is fully observed by all agents. Additionally, the advantages of joint optimization that relieves non-stationary problems and independent experience replay buffer that reduces gradient dependence make JOIE better-learning performance. The improvement is slight on $domain = 2$, but remarkable and impressive as the increasing sizes of the domains. Besides, multi-agent-based models outperform H-DQN, indicating that the proposed collaborative multi-agent framework, which decomposes the joint action space and is led each part by a different agent, can alleviate the exploration obstacles brought by large action space without human knowledge. Finally, DQN is consistently the worst, which is not surprising since it explores and learns from a flat and large action space without any guidance. Noticed that, the performance of DQN increase as the number of domains decreases, which depicts that the growth of action space hinders the learning speeds of RL agent. Meanwhile, as illustrated in Table. 1, the comparison results of Turn and Reward are consistent with that of Succ.

Results of 3-agent based Models Figure 3 shows the learning curves of 3-agent based models. It can be seen that JOIE3 learns faster and performs

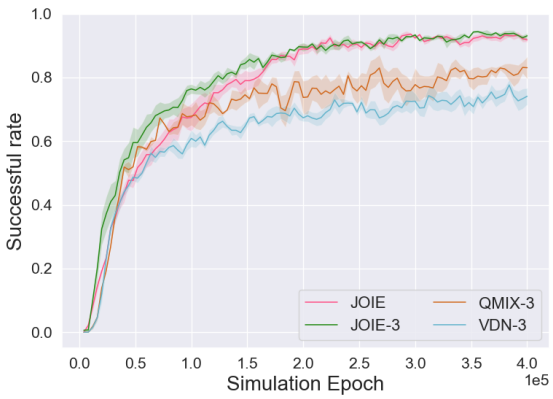


Figure 3: Learning curves of 3-agent based dialog agents trained on all domains.

significantly better with a clear margin compared with VDN3 and Qmix3, which depicts that the decentralized policy with joint optimization and independent experience replay buffer is more capable of and robust to dialog policy learning. JOIE3 factors the concatenated intent-slot action space and assigns them to two agents, which further reduces the action space and balance load for each agent. As a consequence, JOIE3 learns faster than JOIE that based on joint intent-slot action space. Moreover, compared with VDN3 applying a simple sum centralization, Qmix3 adopts a trainable network centralization and achieves better performance.

4.3 Human Evaluation

User simulators are not sufficient to fully mimic the complexity of real users (Dhingra et al., 2017), therefore human evaluation is given to further assess the feasibility of JOIE in real scenarios. we deploy the agents in Figure 2 and 3 to interact with human users in 2-agent based models and 3-agent based models [1] trained on all (seven) domains for 2.0×10^5 simulation epochs.

In each evaluation session, each human user is assigned with a goal sampled goal and instructed to communicate with a randomly selected agent to achieve the goal. Users can end the session at any time if the agent Keeps repeating or they believe the dialog is going to be a failure. At the end of each session, users are required to give explicit feedback on whether the dialog succeeded with all the user constraints satisfied. Moreover, evaluators rate the dialog session on a scale from 1 to 5 about the quality (5 is the best, 1 is the worst). We collect 50 dialogues for each agent. The results are listed in Table 2, which reflects JOIE of both 2-agent based and 3-agent based models perform consistently better than other baselines, which is consistent with what we have observed in simulation evaluation.

Table 2: Human evaluation results on 2-agent based policy models and 3-agent based policy models trained on all domains for 2.0×10^5 simulation epochs. Succ. denotes success rate.

Model	2-agent based		3-agent based	
	Succ.↑	Rating↑	Succ.↑	Rating↑
DQN	0.02	0.06	\	\
H-DQN	0.76	3.68	\	\
JOIE	**0.88**	**4.52**	**0.90**	**4.60**
VDN	0.76	3.60	0.68	3.12
QMIX	0.78	3.82	0.76	3.64

5 Conclusion and Future Work

We presented JOIE, a generally applicable collaborative multi-agent framework for policy learning. It factors action space and learning each part by a different agent with joint optimization and independent experience replay. The experiment results of the simulation show that the proposed agents are efficient and effective in multi-domain with large action space settings.

Directions of future work include: (1) extending JOIE to multi-action policy. (2) improving JOIE with demonstration.

Acknowledgments

We appreciate some insightful comments from the anonymous reviewers; they have helped us improve this paper a lot. The research described in this paper is partially supported by Hong Kong RGCGRF #14204118 and Hong Kong RSFS #3133237.

References

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz–a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*.

Caroline Claus and Craig Boutilier. 1998. The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI/IAAI*, 1998(746-752):2.

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017.

[1] For the time and cost consideration, the experiments are only conducted on all (seven) domains.

Towards end-to-end reinforcement learning of dialogue agents for information access. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 484–495.

Junling Hu and Michael P Wellman. 2003. Nash q-learning for general-sum stochastic games. *Journal of machine learning research*, 4(Nov):1039–1069.

Megha Jhunjhunwala, Caleb Bryant, and Pararth Shah. 2020. Multi-action dialog policy learning with interactive human teaching. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 290–296.

Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Xiang Li, Yaoqin Zhang, Zheng Zhang, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, et al. 2019a. Convlab: Multi-domain end-to-end dialog system platform. *arXiv preprint arXiv:1904.08637*.

Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Zheng Zhang, Yaoqin Zhang, Xiang Li, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, and Jianfeng Gao. 2019b. ConvLab: Multi-domain end-to-end dialog system platform. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 64–69, Florence, Italy. Association for Computational Linguistics.

Jinchao Li, Baolin Peng, Sungjin Lee, Jianfeng Gao, Ryuichi Takanobu, Qi Zhu, Minlie Huang, Hannes Schulz, Adam Atkinson, and Mahmoud Adada. 2020a. Results of the multi-domain task-completion dialog challenge. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence, Eighth Dialog System Technology Challenge Workshop*.

Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. 2017. End-to-end task-completion neural dialogue systems. *arXiv preprint arXiv:1703.01008*.

Ziming Li, Julia Kiseleva, and Maarten de Rijke. 2020b. Rethinking supervised learning and reinforcement learning in task-oriented dialogue systems. *arXiv preprint arXiv:2009.09781*.

Zachary Lipton, Xiujun Li, Jianfeng Gao, Lihong Li, Faisal Ahmed, and Li Deng. 2018. Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Bing Liu and Ian Lane. 2017. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 482–489. IEEE.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.

Alexandros Papangelis, Yi-Chia Wang, Piero Molino, and Gokhan Tur. 2019. Collaborative multi-agent dialogue model training via reinforcement learning. *arXiv preprint arXiv:1907.05507*.

Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, Yun-Nung Chen, and Kam-Fai Wong. 2018a. Adversarial advantage actor-critic model for task-completion dialogue policy learning. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6149–6153. IEEE.

Baolin Peng, Xiujun Li, Jianfeng Gao, Jingjing Liu, and Kam-Fai Wong. 2018b. Deep dyna-q: Integrating planning for task-completion dialogue policy learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 2182–2192.

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. *arXiv preprint arXiv:1704.03084*.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 4295–4304. PMLR.

Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Continuously learning neural dialogue management. *arXiv preprint arXiv:1606.02689*.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. 2017. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*.

Ryuichi Takanobu, Runze Liang, and Minlie Huang. 2020. Multi-agent task-oriented dialog policy learning with role-aware reward decomposition. *arXiv preprint arXiv:2004.03809*.

Huimin Wang, Baolin Peng, and Kam-Fai Wong. 2020. Learning efficient dialogue policy from demonstrations through shaping. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6355–6365.

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. *arXiv preprint arXiv:1702.03274*.

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.

Qi Zhu, Zheng Zhang, Yan Fang, Xiang Li, Ryuichi Takanobu, Jinchao Li, Baolin Peng, Jianfeng Gao, Xiaoyan Zhu, and Minlie Huang. 2020. Convlab-2: An open-source toolkit for building, evaluating, and diagnosing dialogue systems. *arXiv preprint arXiv:2002.04793*.

Table 3: The data annotation schema.

	#domains = 2	#domains = 4	#domains = 7
Domain	Restaurant, Hotel, Booking	Restaurant, Hotel, Booking, Attraction, Taxi	Attraction, Hospital, Booking, Hotel, Restaurant, Taxi, Train, Police, General
Intent	Welcome, Greet, Bye, Reqmore, Inform, Request, Book, OfferBooked, NoBook, NoOffer, Recommend, OfferBook, Select		
Slot	Name, none, Area, Choice, Type, Price, Addr, Leave, Food, Phone, Stars, Day, Post, Car, Arrive, Internet, Parking, Dest, Depart, Fee, Ref, Id, People, Time, Ticket, Stay, Open, Department		

A Data Annotation Schema

Table. 3 lists all annotated dialog domains, intents, and slots for MultiWoz at a different number of domains in detail. Noted that, we didn't count the "General" and the "Booking" as a domain for they cannot define a task independently.

B User simulator

During training, the simulator initializes with a goal and takes system acts as input and outputs user acts with reward, which is set as -1 for each turn, and a positive $(2 \cdot T)$ for successful dialog or a negative of $-T$ for failed one, where T (set as 40) is the maximum number of turns in each dialog. A dialog is considered successful only if the agent helps the user simulator accomplish the goal and satisfies all the user's search constraints (Wang et al., 2020).

C Hyperparameters and Implementation

Set $m \in 2, 4, 9$ as the numbers of domains. We adopt 2-layer MLP with 100 hidden dimensions and Relu as the activation function for all m. Inputting state with dimension as 393, DQN's output dimension is $m * 364$. Where 364 is the number of action concatenating intent and slot. 2-agent based models with combined intent and slot action space, i.e. H-DQN, VDA, Qmix, JOIE, utilize two networks with different output heads of m and 364 dimensions. Noted that, VDA, Qmix, JOIE share input, and hidden layers. 3-agent based models with separated domain, intent, and slot action space, i.e. VDA3, Qmix3, JOIE3, apply three different output heads of $m, 13$, and 28 dimensions and share input and hidden. ϵ-greedy is utilized for policy exploration. We set the discount factor as $\gamma = 0.9$. The target networks are updated at every 1000 training epochs. To mitigate warm-up issues, We apply the rule-based agent of ConvLab (Lee et al., 2019a) to provide experiences at the beginning, the warm_start epoch for all agents is 1000. The learning rate is set as 0.001 for DQN, 0.0005 for JOIE3, and 0.00005 for the other models. The decay rate and step size are 0.95 and 1000.

D Algorithms

Algorithm 1 outlines the full procedure for training multi-agent-based dialogue policies based on joint optimization and independent experience replay buffers.

Algorithm 1 JOIE for dialog policy learning

Input: $N, Z, \epsilon, \theta_d, \theta_i, \theta_s, D_d, D_i, D_s, \gamma$,
Output: $Q_{\theta_d}(s, a^d), Q_{\theta_i}(s||a^d, a^i), Q_{\theta_s}(s||a^d||a^i, a^s)$.

```
init experience replay D_d, D_i, D_s as empty.
init Q_{θ_d}, Q_{θ_i}, Q_{θ_s}, Q_{θ'_d}, Q_{θ'_i}, Q_{θ'_s}) with θ_d = θ'_d, θ_i = θ'_i, and θ_s = θ'_s.
for n=1:N do
    start dialog simulator and get state s.
    while s is not terminal do
        with probability ϵ select a random action a^d.
        otherwise a^d = argmax_a Q_{θ_d}(s, a).
        with probability ϵ select a random action a^i.
        otherwise a^i = argmax_a Q_{θ_i}(s||a^d, a).
        with probability ϵ select a random action a^s.
        otherwise a^s = argmax_a Q_{θ_s}(s||a^d||a^i, a).
        execute (a^d, a^i, a^s), obtain next state s', reward r.
        set (a^d)' as none and (a^i)' as none, store transition (s, a^d, s', r) in D_d, (s, a^d, a^i, s', (a^d)', r) in D_i and update the last transition (a^d)' = a^d, (s, a^d, a^i, a^s, s', (a^d)', (a^i)', r) in D_s and update the last transition (a^d)' = a^d, (a^i)' = a^i.
    end while
    update the last transition in D_i, D_s.
    Sample batch1 of (s, a^d, r, s') from D_d
    Sample batch2 of (s, a^d, a^i, (a^d)', r, s') from D_i
    Sample batch3 of (s, a^d, a^i, a^s, s', (a^d)', (a^i)', r) from D_s
    update Q_{θ_d}, Q_{θ_i}, and Q_{θ_s} via minibatch Q-learning according to gradient of equ.7.
    every Z steps reset Q_{θ_d} = Q_{θ'_d}, Q_{θ'_i} = Q_{θ'_i}, and Q_{θ'_s} = Q_{θ'_s}.
end for
```