applied sciences

MDPI

Article

Cooperative Multi-Agent Reinforcement Learning with Conversation Knowledge for Dialogue Management

Shuyu Lei *, Xiaojie Wang and Caixia Yuan

Center for Intelligence of Science and Technology (CIST), Beijing University of Posts and Telecommunications, Beijing 100876, China; xjwang@bupt.edu.cn (X.W.); yuancx@bupt.edu.cn (C.Y.)

* Correspondence: leishuyu@bupt.edu.cn

Received: 25 March 2020; Accepted: 9 April 2020; Published: 15 April 2020

Abstract: Dialogue management plays a vital role in task-oriented dialogue systems, which has become an active area of research in recent years. Despite the promising results brought from deep reinforcement learning, most of the studies need to develop a manual user simulator additionally. To address the time-consuming development of simulator policy, we propose a multi-agent dialogue model where an end-to-end dialogue manager and a user simulator are optimized simultaneously. Different from prior work, we optimize the two-agents from scratch and apply the reward shaping technology based on adjacency pairs constraints in conversational analysis to speed up learning and to avoid the derivation from normal human-human conversation. In addition, we generalize the one-to-one learning strategy to one-to-many learning strategy, where a dialogue manager can be concurrently optimized with various user simulators, to improve the performance of trained dialogue manager. The experimental results show that one-to-one agents trained with adjacency pairs constraints can converge faster and avoid derivation. In cross-model evaluation with human users involved, the dialogue manager trained in one-to-many strategy achieves the best performance.

Keywords: dialogue management; user simulation; reward shaping; conversation knowledge; multi-agent reinforcement learning

1. Introduction

A task-oriented dialogue system can help people accomplish specific goals, such as booking a hotel, seeking a restaurant information. A typical text-based task-oriented dialogue system mainly comprises three parts—Natural Language Understanding (NLU), Dialogue Management (DM), and Natural Language Generation (NLG). DM plays a vital role which infers dialogue state from NLU and provides appropriate action for NLG, and it has attracted much attention in recent years.

Recently, reinforcement learning has been widely studied as a data-driven approach for modeling DM [1–9], where a state tracker maintains dialogue states and a policy model chooses a proper action according to the current dialogue state. In most recent studies [4–9] on task-oriented dialogue tasks, Deep Reinforcement Learning (DRL) was utilized to train the policy model in order to achieve maximum long-term reward through interacting with a manual user simulator. To this end, most of the studies need the additional development of a user simulator in task-oriented dialogue system.

To address the time-consuming development of simulator policy issue, we propose a Multi-Agent Dialogue Model (MADM) where an end-to-end dialogue manager cooperates with a user simulator to fulfill the dialogue task. Since user simulator is treated as one agent in multi-agent, the simulator policy can be optimized in an automatic manner rather than laboring development. Different from prior work [10], we optimize the cooperative policies concurrently via multi-agent reinforcement learning

from scratch without supervised initializing process. For user simulator reward function, we use the reward shaping technique [11] based on the adjacency pairs in conversational analysis [12] to make the simulator learn real user behaviors quickly. In addition, we generalize the one-to-one learning strategy to one-to-many learning strategy where a dialogue manager cooperates with various user simulators to improve the performance of trained dialogue manager. We obtain these various user simulators through changing the adjacency pairs settings, and then we mixture them with a dialogue manager to optimize the cooperative policies via multi-agent reinforcement learning.

Compared with MADM without the constraints, MADM trained with adjacency pairs constraints can converge faster and avoid derivation from normal human-human conversation. The experimental results also show that the dialogue manager trained with one-to-many strategy achieves the best performance in cross-model evaluation with human users involved. To summary, our main contributions in this work are three-fold:

1. We propose an MADM to optimize the cooperative policies between an end-to-end dialogue manager and a user simulator concurrently from scratch.
2. We apply reward shaping technique based on adjacency pairs to user simulator to speed learning and to help the MADM generate normal human-human conversation.
3. We further generalize the one-to-one learning strategy to one-to-many learning strategy to improve the performance for trained dialogue manager.

The rest of the paper is organized as follows—Section 2 gives an overview of related work. Section 3 describes the MADM model in detail. Section 4 discusses the experimental results and evaluations. Section 5 gives the conclusive discussions and the description of future work.

2. Related Work

Data-driven DM has become an active research area in the field of task-oriented dialogue system. In recent years, a lot of promising studies [1,2,4,7–9] worked on the policy model in dialogue system pipeline. Meanwhile, some studies [13–15] built the DM and NLU into an end-to-end model. In the above studies, the dialogue policy was optimized with a user simulator as a trial-and-error manner in reinforcement learning. However, the development of a user simulator was complex and it took considerable time to built an appropriate user policy. Additionally, some studies [4,5,14,16] relied on considerable supervised data. Reference [16] proposed an end-to-end model by jointly training NLU and DM with supervised learning. References [4,5,14] applied the demonstration data to speed up the convergence in a supervised manner. Preparing such supervised data is also laborious. Although some studies [3,17] could optimize the policy model via on-line human interaction, these methods required considerable human interaction. Meanwhile, the initial performance was still relatively poor, which could impact negatively on the user experience. Different from the above studies, the dialogue management in our framework is optimized from scratch without any laborious preparation for supervised data and development of user policy.

As the user simulator plays a vital role in reinforcement learning for optimizing dialogue policy, the studies on the user simulator also received a lot of attention. References [18–24] utilized the data-driven approach to develop the user simulator. However, such statistic-based methods required a lot of corpus. Once the training data were not sufficient, the data-driven simulator could only produce a simplex response. Dialogue management trained with such simplex simulator might converge to a solution with poor generalization performance. In addition, the obtained policy was uncontrollable with statistic-based methods. Thus, an alternative approach was based on agenda rules. Reference [25] proposed an agenda-based approach that does not necessarily need training data but can be trained in case such data are available. This agenda-based simulator was realistic enough to successfully test many DRL algorithms [6] and train a dialogue policy. However, the developer must maintain the rules operating on agenda, working as simulation policy, with domain expertise. Different from above

studies, user simulator in our framework is optimized from scratch without the need of pre-defined rules or dialogue corpus.

To address the time-consuming development for simulator policy, recent studies [10,26,27] proposed a one-to-one dialogue model where a dialogue manager and a user simulator were optimized concurrently. Different from the above studies, our proposed MADM applies the reward shaping technique [11] based on the adjacency pairs in conversational analysis [12], which can help the cooperative policies learn from scratch quickly. By the method of reward shaping, our proposed MADM avoids running a learning algorithm multiple times in a study [26] and collects the corpora in studies [10,27].

Recently, multi-agent reinforcement learning has been applied in many interesting research areas. References [28,29] proposed a cooperative 'image guessing' game between two agents – Q-BOT and A-BOT– who communicate in natural language dialog so that Q-BOT can select an unseen image from a lineup of images. References [30,31] showed it was possible to train a multi-agent model for negotiation where agents with different goals attempt to agree on common decisions. Reference [32] pointed out that a competitive multi-agent environment trained with self-play could produce behaviors that were far more complex than the environment itself. Different from the above studies, we use the multi-agent reinforcement learning to model the cooperation between dialogue manager and user simulator.

3. Model

3.1. Notation

We consider a cooperative multi-agent reinforcement learning as a Decentralized Partially Observable Markov Decision Processes (Dec-POMDP) [33] defined with a tuple $(\alpha, \mathcal{S}, \{\mathcal{A}_i\}_{i\in\alpha}, \mathcal{T}, \{\mathcal{O}_i\}_{i\in\alpha}, \mathcal{Z}, \{\mathcal{R}_i\}_{i\in\alpha})$, where α is a set of n agents, $\mathcal{S}$ is a set of states of the world and the possible joint configuration of all the agents, $\mathcal{A}_i$ is a set of actions for agent i, the joint action space are defined as $\mathcal{A} = \mathcal{A}_1 \times ... \times \mathcal{A}_n$, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is a state transition function, $\mathcal{O}_i$ is a set of observations for agent i, the joint observation space are denoted as $\mathcal{O} = \mathcal{O}_1 \times ... \times \mathcal{O}_n$, $\mathcal{Z} : \mathcal{S} \times \mathcal{A} \times \mathcal{O} \to [0, 1]$ is an observation function, $\mathcal{S}_i : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is a reward function for agent i. For the cooperative multi-agent reinforcement learning, each agent i has the equal reward in every time step t. Each agent i chooses its own actions according to the policy function $\pi_i : \mathcal{O}_i \times \mathcal{A}_i \to [0, 1]$. Each agent i aims to maximize its own long-term discounted reward $R_i = \sum_{t=0}^{T} \gamma^t r_{i,t}$, where γ is a discount factor and T is the time horizon.

3.2. Multi-Agent Dialogue Model (MADM)

We propose an MADM where a dialogue manager cooperates with a user simulator to fulfill the dialogue task based on cooperative multi-agent reinforcement learning. The entire architecture is illustrated in Figure 1. The basic MADM has two agents: a dialogue manager and a user simulator. This basic MADM can be generalized to MADM with multiple agents—a dialogue manager and various user simulators. The dialogue manager takes the historical dialogue sequence as input and then produces the selected action. The user simulator takes the action from the dialogue manager and then produces a user utterance back to dialogue manager. The dialogue manager and the user simulator are described in detail, respectively, as follows.

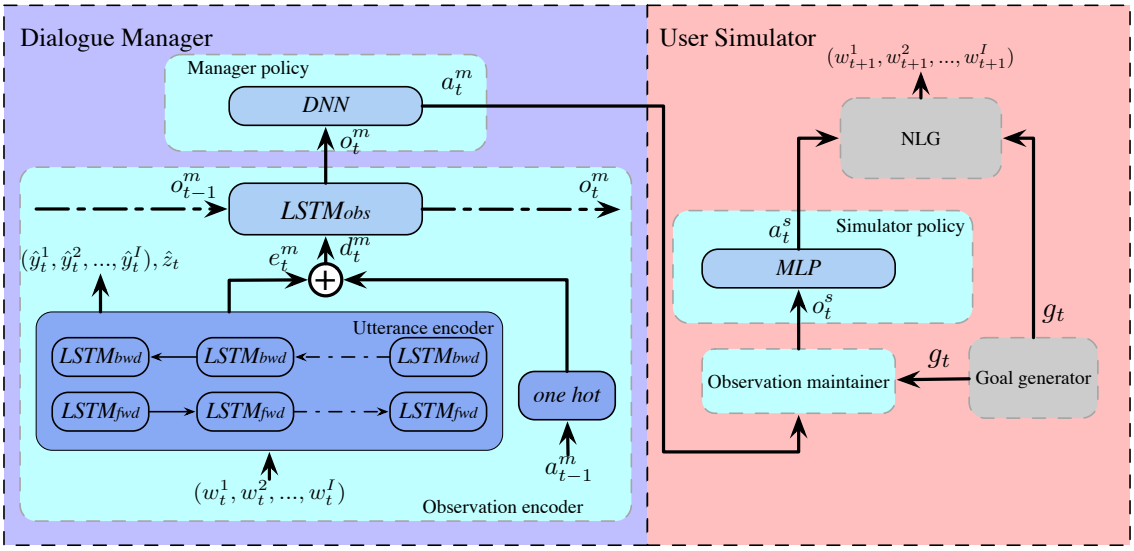

Figure 1. The cooperative multi-agent dialogue model between dialogue manager and user simulator.

3.2.1. Dialogue Manager

Dialogue manager consists of two parts: an observation encoder and a manager policy as shown in Figure 1. The observation encoder is employed to map historical dialogue sequence to observation representation. As some slot dependent actions (e.g., *confirm*()) need to combine with slot values from user utterances to make up an integral action, observation encoder also produces the slot values from user utterance through slot filling and intent recognition. The manager policy is applied to map the observation representation to a selected action for responding to user simulator. Observation encoder and manager policy are described in detail, respectively, as follows.

Observation encoder: the historical dialogue sequence $h_t = [a_0^m, u_1, ..., a_{t-1}^m, u_t]$ is encoded to an observation representation o_t^m, meanwhile, the slot values information y^t and the intent recognition information z^t are output, where a_{t-1}^m denotes the selected action from manager in time step $t-1$, $u_t = [w_t^1, w_t^2, ..., w_t^I]$ denotes the user utterance in time step t, w_t^i denotes the i-th word (or i-th character in Chinese) in the user utterance u_t, and $\hat{y}_t = [\hat{y}_t^1, \hat{y}_t^2, ..., \hat{y}_t^I]$ denotes the slot label information on user utterance u_t. To this end, a hierarchical recurrent neural network (HRNN) is applied to model observation encoder. In the bottom layer of HRNN, a bidirectional LSTM [34] with attention pooling is employed to obtain the sentence representation e_t^m for user utterance u_t, which is computed as follows:

$$\overrightarrow{c}_t^i = LSTM_{fwd}(\overrightarrow{c}_t^{i-1}, e(w_t^i)) \quad (1)$$

$$\overleftarrow{c}_t^i = LSTM_{bwd}(\overleftarrow{c}_t^{i+1}, e(w_t^i)) \quad (2)$$

$$e_t^i = \sum_{i=1}^{I} \alpha_i [\overrightarrow{c}_t^i \oplus \overleftarrow{c}_t^i] \quad (3)$$

$$\alpha_i = \frac{\exp q_t^i}{\sum_{i=1}^{I} \exp q_t^i} \quad (4)$$

$$q_t^i = g([\overrightarrow{c}_t^i \oplus \overleftarrow{c}_t^i]), \quad (5)$$

where $\overrightarrow{c}_t^i$ and $\overleftarrow{c}_t^i$ are the outputs of forward and backward LSTM in bottom layer of HRNN, respectively, e_t^i denotes the embedding of word w_t^i, $\oplus$ is the concatenation operator, α_i is the attention

weights, and g is a feed-forward neural network. The bidirectional LSTM also outputs the slot values information $\hat{y}_t$ and the intent recognition information $\hat{z}^t$, which is computed as follows:

$$\hat{y}_t^i = \arg\max_l(softmax(\overrightarrow{c}_t^i \oplus \overleftarrow{c}_t^i)) \tag{6}$$

$$\hat{z}_t = \arg\max_k(softmax(\overrightarrow{c}_t^I \oplus \overleftarrow{c}_t^0)), \tag{7}$$

where l denotes the set of slot labels and k denotes the set of intent labels. In top layer of HRNN, a forward LSTM is applied to integrate the last observation representation o_{t-1}^m, last manager action a_{t-1}^m and current sentence representation e_t^m into current observation representation o_t^m, which is computed as follows:

$$d_t^m = e_t^m \oplus o(a_{t-1}^m) \tag{8}$$

$$o_t^m = LSTM_{obs}(o_{t-1}^m, d_t^m), \tag{9}$$

where d_t^m is the concatenation of sentence representation e_t^m and last action representation $o(a_{t-1}^m)$, and $o(a_{t-1}^m)$ is a one-hot vector with the corresponding action position set to 1.

Manager policy: the observation representation o_t^m is projected to the selected action a_t^m. To this end, a deep neural network (DNN) is applied to model manager policy, which is computed as follows:

$$\pi^m(a_t^m|o_t^m) = softmax(DNN(o_t^m)), \tag{10}$$

where policy function $\pi^m(a_t^m|o_t^m)$ is a probability distribution on the action space. The selected action a_t^m is drawn from the distribution $\pi^m(a_t^m|o_t^m)$. For convenience, $\pi^m(a_t^m|o_t^m;\theta^m)$ is denoted as the policy function of dialogue manager, where θ^m are the parameters of the manager policy.

3.2.2. User Simulator

User simulator is composed of four parts: a simulator observation maintainer, a goal generator, a simulator policy, and an NLG as shown in Figure 1. The observation maintainer is applied to obtain the observation representation for user simulator. The goal generator is used to produce the user goal (e.g., slot value) and simulate the goal change during a dialogue. The simulator policy is applied to map the observation representation to a selected action for generating a user utterance. The NLG is applied to generate the next user utterance to dialogue manager. The four parts of user simulator are described in detail, respectively, as follows.

Observation maintainer: the observation representation o_t^s is a concatenated vector composed of three parts: an embedding $o(a_t^m)$ for manager action a_t^m, a binary variable b_t that indicates whether the slot value in manager action a_t^m is null, and an indicative vector v_t that denotes which type of slot value in confirm-action received from manager is different from user goal g_t in time step t.

Goal generator: the user goal is generated at the start of the dialogue by sampling the candidate slot values uniformly. As the initial goal may change in a real user dialogue, the variation of user goals are also simulated during the interaction. For each session, the user goals are sampled from the candidate slot values randomly at the beginning of the dialogue, meanwhile, an indicative vector c_c, which counts the number of variations for each slot, is set to be a zeroes vector. This indicative vector c_c is used to limit the number of variations for each slot to avoid overly complex conversations. In each turn, a variation probability p_v is sampled from $[0, 1]$ randomly, if this variation probability p_v is bigger than threshold probability p_{th}, then a random slot is selected to change the corresponding value to another one from candidate slot values. Once a slot value is changed, the corresponding value of variation slot in indicative vector c_c is added 1. If the number of variations for some slots exceed the limitation number, those slots will not be changed, even though the variation probability p_v is bigger than threshold probability p_{th}.

Simulator policy: the observation representation o_t^s is mapped to the selected action a_t^s. To this end, a multi-layer perceptron (MLP) is applied to model simulator policy, which is computed as follows:

$$\pi^s(a_t^s|o_t^s) = softmax(MLP(o_t^s)), \tag{11}$$

where policy function $\pi^s(a_t^s|o_t^s)$ is a probability distribution on the action space. The selected action a_t^s is drawn from the distribution $\pi^s(a_t^s|o_t^s)$. For convenience, $\pi^s(a_t^s|o_t^s;\theta^s)$ is denoted as policy function of user simulator, where θ^s are the parameters of the simulator policy.

NLG: the selected action a_t^s is projected to next user utterance u_{t+1} for replying to dialogue manager. A template-based NLG is used to produce such user utterances. The responding template is drawn from a set of pre-defined templates according to the selected action a_t^s. To assure the generalization and expressiveness, the templates are delexicalized by replacing concrete slot values with their slot names. For some slot dependent actions (e.g., *inform*()), the drawn template is lexicalized with the goal slot values to generate the final user utterance. An example of user utterance generation is shown in Figure 2, where B-loc, I-loc and O denote the slot labels of the beginning character of a location, inter character of a location and other characters, respectively.

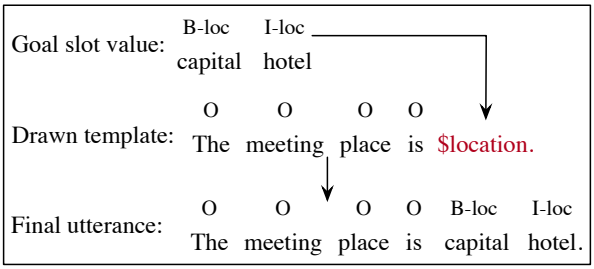

Figure 2. An example of user utterance generation.

3.3. Cooperative Training

Policy gradient: the policy gradient is applied to compute an estimate of the gradient of policy parameters in order to maximize the long-term discounted reward. In a cooperative dialogue, the gradient of manager policy and simulator policy are denoted as follows:

$$\nabla_{\theta^m} J(\theta^m) = \mathbb{E}_{\pi^m,\pi^s}[A^m(a^m,o^m)\nabla_{\theta^m} \log \pi^m(a^m|o^m)] \tag{12}$$

$$\nabla_{\theta^s} J(\theta^s) = \mathbb{E}_{\pi^m,\pi^s}[A^s(a^s,o^s)\nabla_{\theta^s} \log \pi^s(a^s|o^s)], \tag{13}$$

where $A^m(a^m,o^m)$ is the advantage function of manager, and $A^s(a^s,o^s)$ is the advantage function of simulator. REINFORCE with a baseline algorithm [35] is applied to estimate the advantage functions. Thus, the advantage function $A^m(a^m,o^m)$ and the advantage function $A^m(a^m,o^m)$ are computed as follows:

$$A^m(a_t^m,o_t^m) = \sum_{j=0}^{J} \gamma^j r_{t+j} - V^{\pi^m}(o_t^m;\phi^m) \tag{14}$$

$$A^s(a_t^s,o_t^s) = \sum_{j=0}^{J} \gamma^j r_{t+j} - V^{\pi^s}(o_t^s;\phi^s), \tag{15}$$

where $V^{\pi^m}(o_t^m;\phi^m)$ is the value function of manager with parameters ϕ^m to estimate the return on observation o_t^m, and $V^{\pi^s}(o_t^s;\phi^s)$ is the value function of simulator with parameters ϕ^s to estimate the return on observation o_t^s. The loss functions of $V^{\pi^m}(o_t^m;\phi^m)$ and $V^{\pi^s}(o_t^s;\phi^s)$ are computed as follows:

$$J(\phi^m) = \frac{1}{2}[A^m(a_t^m, o_t^m)]^2 \tag{16}$$

$$J(\phi^s) = \frac{1}{2}[A^s(a_t^s, o_t^s)]^2. \tag{17}$$

The value function $V^{\pi^m}(o_t^m;\phi^m)$ and policy function $\pi^m(a_t^m|o_t^m;\theta^m)$ share the same parameters, meanwhile, the slot filling and intent recognition are optimized in a supervised manner jointly. To this end, the total loss function of dialogue manager is computed as follows:

$$J_r(\theta^m) = -A^m(a^m, o^m)\nabla_{\theta^m}\log\pi^m(a^m|o^m) + \frac{1}{2}[A^m(a^m, o^m)]^2 \tag{18}$$

$$J_s(\theta^m) = \sum_{t=1}^{T}\sum_{i=1}^{I}\hat{y}_t^i \log y_t^i + \sum_{t=1}^{T}\hat{z}_t \log z_t \tag{19}$$

$$J_w(\theta^m) = (1-\lambda)J_r(\theta^m) + \lambda J_s(\theta^m), \tag{20}$$

where $\lambda \in (0, 1]$ is a balance coefficient. Similar to dialogue manager, the value function $V^{\pi^s}(o_t^s;\phi^s)$ and policy function $\pi^s(a_t^s|o_t^s;\theta^s)$ share the same parameters in user simulator. The total loss function of user simulator is computed as follows:

$$J_w(\theta^s) = -A^s(a^s, o^s)\nabla_{\theta^s}\log\pi^s(a^s|o^s) + \frac{1}{2}[A^s(a^s, o^s)]^2. \tag{21}$$

The two total-loss functions are optimized cooperatively after a complete dialogue. In this way, the dialogue manager and the user simulator are optimized cooperatively and simultaneously. The alternate training method was tried to optimize dialogue manager and user simulator, and empirical results show that alternate training method (every 10 training steps alternately) has slower convergence than joint training method and achieves the same performance with training jointly.

Above all, the dialogue manager and the user simulator are optimized cooperatively in a one-to-one manner. To improve the dialogue manager generalization performance, this one-to-one cooperation is generalized to one-to-many cooperation where a dialogue manager cooperates with various user simulators. These various user simulators are obtained through changing the settings of adjacency pairs as described in the next paragraph. For one training step, dialogue manager interacts with one user simulator to fulfill a complete dialogue, then the dialogue manager and the current simulator are optimized via one-to-one training. For next training step, dialogue manger changes to anther simulator to learn the cooperative policies. In this way, the dialogue management and the various user simulators are optimized in a one-to-many manner alternately. We tried to use multi one-to-one parallelly then share the gradient of dialogue manager, and empirically observed that sharing gradient optimization is slower than learning one-by-one.

Reward shaping based on adjacency pairs: In cooperative multi-agent reinforcement learning, each agent has the same reward for every time step. The naive reward function is assigned as follows:

- Manager reward $r(s_{t-1}, a_{t-1}^m, s_t)$ and simulator reward $r(s_{t-1}, a_{t-1}^s, s_t)$ are both $+1$, if s_t is a successful completed state.
- Manager reward $r(s_{t-1}, a_{t-1}^m, s_t)$ and simulator reward $r(s_{t-1}, a_{t-1}^s, s_t)$ are both -1, if s_t is not a successful completed state until the maximum length T in a dialogue.
- Manager reward $r(s_{t-1}, a_{t-1}^m, s_t)$ and simulator reward $r(s_{t-1}, a_{t-1}^s, s_t)$ are both -0.01 in otherwise.

This credit-assignment approach is sparse and delayed when a successful cooperative dialogue between dialogue manager and user simulator has a long trajectory. In cold start situation, as the

initial cooperative polices are nearly random, the successful dialogue with a long trajectory is easier to be generated than one with a short trajectory. This credit-assignment approach leads to a slow convergence. To alleviate this problem, we use the reward shaping technique [11] based on the adjacency pairs in conversational analysis [12] to substitute the reward in user simulator. The reward based on the adjacency pairs is assigned as follows:

- Simulator reward $r(s_{t-1}, a^s_{t-1}, s_t)$ is -0.01, if s_t is a non-terminal state and the action pair $[a^m_{t-1}, a^s_{t-1}]$ does not belong to the set of adjacency pairs.
- Simulator reward $r(s_{t-1}, a^s_{t-1}, s_t)$ is r_s, if s_t is a non-terminal state and the action pair $[a^m_{t-1}, a^s_{t-1}]$ does not belong to the set of adjacency pairs, where r_s is the shaping reward greater than -0.01.
- Manager reward $r(s_{t-1}, a^m_{t-1}, s_t)$ and simulator reward $r(s_{t-1}, a^s_{t-1}, s_t)$ are both $+1$, if s_t is a successful completed state.
- Manager reward $r(s_{t-1}, a^m_{t-1}, s_t)$ and simulator reward $r(s_{t-1}, a^s_{t-1}, s_t)$ are both -1, if s_t is not a successful completed state until the dialogue reaches maximum length T in a dialogue.
- Manager reward $r(s_{t-1}, a^m_{t-1}, s_t)$ is -0.01, if s_t is a non-terminal state.

Through changing the set of adjacency pairs, various user simulators can be obtained. For non-shaped reward setting, each agent has the equal reward every time step. For shaped reward setting, each agent aims to maximize its own long-term discounted reward.

4. Experiment

To assess the performance, cross-model evaluation [36] is applied that is, training on one simulator and testing on the other. In our cross-model evaluation, human users also take part in the test for different dialogue managers. The evaluation is happened on Chinese meeting room booking tasks. It is worth nothing that our proposed framework can be directly utilized on English tasks by substituting Chinese characters to English words as inputs.

4.1. Dataset

The dataset was collected from 300 human-human dialogues on booking Chinese meeting room task. The average length of collected dialogues is approximately 16 turns. For the NLG in user simulator, 255 pre-defined templates and 240 slot values are extracted from collected dialogues. The dialogue manager consists of 7 dialogue acts and 3 slots and the user simulator consists of 10 dialogue acts, as shown in Table 1.

Table 1. lists all dialogue acts in details.

	Dialogue Acts
Dialogue manager	ask_date,ask_location,ask_attendance, confirm_date,confirm_location,confirm_attendance,bye
User simulator	inform_date,inform_location,inform_attendance,update_date, update_location,update_attendance,affirm,deny,error,hello

4.2. Cross-Model Evaluation with Human Users Involved

4.2.1. Users for Cross-Model Evaluation

To access the performance on different dialogue managers, simulated users and human users take part in the cross-model evaluation.

A group of user simulators (Group-S): This group of user simulators is obtained through changing the settings of adjacency pairs and is optimized with the dialogue manager in MADM as one-to-many strategy via multi-agent reinforcement learning. The Group-S is composed of five different simulators: all-simulator where all the types of adjacency pairs is applied to reward shaping,

ask-simulator where only ask-action adjacency pairs (e.g., ask_loc() to inform_loc()) is applied to reward shaping, confirm-simulator where only confirm-action adjacency pairs (e.g., confirm_loc() to affirm()) is applied to reward shaping, bye-simulator where only bye-action adjacency pairs (e.g., bye() to bye()) is applied to reward shaping and naive-simulator where no adjacency pairs is applied. The shaping reward r_s is set to $+0.01$. The probability of simulating goal change is set to 0.5. Each slot is limited to change once to avoid overly complex conversations. For the NLG, the collected pre-defined templates are used to generate the user utterance through lexicalization as described in Section 3.2.2. Different dialogue managers are tested with each simulator in Group-S through interacting 200 episodes.

A rule-based user simulator (Rule-S): This simulator is developed according to the mode proposed in Reference [25,37]. The naive reward function is used in Section 3.3. The same settings in Group-S is used for goal generator and NLG. Different dialogue managers are tested with this Rule-S through interacting 200 episodes.

Human Users: 25 graduate volunteers are recruited to conduct human users test. Comparing different model subjectively on human users always suffers from unfairness and human user may fit in the system gradually. Thus, human users test is conducted in a paralleled manner and is evaluated in objective assessment whether the system can help users accomplish tasks or not. Before testing, the specific user goals are allocated to each users. In the guide of the same allocated goal, the human users use the same natural language to interact with different dialogue managers. Each of the volunteers conducts two parallel tests on different dialogue managers.

4.2.2. Dialogue Managers for Cross-Model Evaluation

To benchmark the dialogue manager from MADM trained as one-to-many strategy, five dialogue managers take part in the cross-model evaluation.

A dialogue manager from MADM trained as one-to-many strategy (M-MADM-OM): This end-to-end dialogue manager is built based on the dialogue manager as described in MADM and optimized with the Group-S concurrently via multi-agent reinforcement learning. The character is used as the model inputs, the size of character embedding is set to 8, the hidden sizes of the LSTM in bottom layer of HRNN and LSTM in bottom layer of HRNN are both set to 16, the sizes of two hidden layers in DNN are both set to 16 and the balance coefficient λ is 0.5.

A dialogue manager trained with Rule-S (Rule-M): This end-to-end dialogue manager is implemented with the same inputs and structures as dialogue manager in MADM and is optimized with the Rule-S through REINFORCE with baseline algorithm.

Yang 2017 [16]: A end-to-end dialogue manager is implemented as those in Reference [16]. The hidden size of the LSTM for NLU and system action prediction are both set to 16. This model is optimized with standard supervised learning.

Zhao 2016 [13]: A end-to-end dialogue manager is implemented with the same inputs and structure as those in Reference [13]. The hidden size of the LSTM is set to 256. The size of hidden layer which maps LSTM output to action is 128. As the model in Reference [13] can only parse a Yes/No answer, we connect this model with additional NLU. This NLU is modeled with a bi-directional LSTM separately. The hidden size of separate bi-directional LSTM is set to 32. This model optimized with REINFORCE with baseline outperforms the one optimized with deep Q-learning after repeated experiments in our dialogue tasks. Thus, REINFORCE with baseline algorithm is used to optimize this model with the Rule-S.

Peng 2018 [9]: A dialogue manager implements a model with the same inputs and structures as dialogue manager in MADM. This dialogue manager is optimized with deep dyna-Q with a world model and a user simulator. The world model is implemented with the same structure as in Reference [9], where the input is the concatenation of an observation representation o_t^m and an embedding of dialogue manager action a_t^m, where the size of hidden layer is set to 16. The user simulator uses the same setting in Rule-S.

4.2.3. Results

The results of the cross-model evaluation on success rate and average turns are shown in Table 2. In Group-S test, M-MADM-OM achieves the best performance. In Rule-S test, although M-MADM-OM does not achieve the best performance, it is only 0.2% lower than Rule-M and Peng 2018 [9]. In human users test, M-MADM-OM achieves the best performance. Above all, our proposed M-MADM-OM achieves the best performance in cross-model evaluation.

Table 2. Cross-model evaluation on Success Rate (SR) and Average Turns (AT).

	Group-S		**Rule-S**		**Human Users**	
	SR	**AT**	**SR**	**AT**	**SR**	**AT**
M-MADM-OM	**0.902**	**18.86**	0.925	17.28	**0.84**	**18.04**
Rule-M	0.582	24.94	**0.945**	**17.04**	0.76	19.56
Yang2017	0.577	25.03	0.860	21.56	0.68	21.08
Zhao2016	0.433	27.77	0.890	20.99	0.68	20.02
Peng2018	0.428	27.86	**0.945**	18.44	0.72	20.32

For the simulators performance, comparing Group-S test with Rule-S test, dialogue managers trained with Rule-S show the bad performance while interacting with Group-S. This phenomenon shows that Group-S may generate some user behaviors that Rule-S are unable to simulate. Comparing Group-S test with human users test, the results of human users are better than Group-S, which means that Group-S generate some user behaviors that human users may not produce. Even so, to our surprise, the Group-S can improve the concurrent dialogue manager performance on human users test.

Since our method applies a dynamic adjusted simulator without extensive involving of human laboring, the built model is more time efficient in a long run, even though it is slower in learning an optimal dialogue manager compared with the one-to-one methods with rule-based user simulator (including the work in Reference [9]). As empirical analysis, we observed that dialogue manager with dynamic adjusted simulator is four hours slower than deep dyna-Q method in Reference [9] as the same experimental settings, finally we obtained the optimized simulator with better generalization ability and without involving any more human efforts.

4.2.4. Good Case Study

Considering the improvement on M-MADM-OM in real scenario, two examples compared between M-MADM-OM and Rule-M are shown in Table 3. The Rule-M may fail in the case that the user always gives irrelevant answer (e.g., system request the number of people and user inform the date of the meeting). On the other hand, the M-MADM-OM can tackle such irrelevant answer and guide the user to inform the rest of slots. This is because the Group-S may generate more user behaviors than Rule-S, and M-MADM-OM can learn more robust policy for real scenario than Rule-M.

Table 3. Two sample dialogue sessions on human users comparing M-MADM-OM with Rule-M dialogue manager (SYS: system, USR: human user).

M-MADM-OM	Rule-M
1 SYS: Hello, where is the meeting?	1 SYS: Hello, where is the meeting?
2 USR:The location is in bupt technology building.	2 USR:The location is in bupt technology building.
3 SYS: Please confirm, the meeting place is in bupt technology building?	3 SYS: Please confirm, the meeting place is in bupt technology building?
4 USR:Yes.	4 USR:Yes.
5 SYS: How many people are there?	5 SYS: How many people are there?
6 USR:The meeting will be held at 10 a.m. on 5 April 2019.	6 USR:The meeting will be held at 10 a.m. on 5 April 2019.
7 SYS: Will the meeting be held at 10 a.m. on 5 April 2019?	7 SYS: How many people are there?
8 USR:Sorry, please change 2 p.m. on 5 April 2019.	8 USR:The meeting will be held at 10 a.m. on 5 April 2019.
9 SYS: Please confirm. the meeting be held at 2 p.m. on 5 April 2019?	9 SYS: How many people are there?
10 USR:Pefect.	.
11 SYS: How many people are there?	.
12 USR:Eight.	.
13 SYS: Is there eight people to participate?	32 SYS: How many people are there?
14 USR:For sure.	
15 SYS: Booked at 14:00 on 5 April 2019 Sixth meeting room at bupt technology building for you, if there are any questions, please call 010-52810888, good bye.	
Success	**Failure:** System does not book the right meeting room within the max turn

4.3. Ablation

The ablation experiments are conducted to evaluate efficiency of the different settings on adjacency pairs for reward shaping and the generalization performance on M-MADM-OM.

4.3.1. Adjacency Pair Performance

Considering reward shaping influence on convergence, the different adjacency pairs settings for reward shaping are compared. There are five settings: all the types of adjacency pairs, only ask-action adjacency pairs, only confirm-action adjacency pairs, only bye-action adjacency pairs and naive reward function. The training curves are shown in Figure 3. These success rate curves are obtained through testing dialogue managers with their respective learning simulator after every 300 training steps. Two settings (i.e., all the types of adjacency pairs and only ask-action adjacency pairs) achieve the best performance on speed up learning.

As the learning from scratch may cause that learned policy deviate from normal human-human conversation, these final dialogue managers are also tested with human users to check whether they deviate from normal human-human conversation or not. The same paralleled test strategy as described in Section 4.2.1 is conducted in human users test. The success rate and average turns are shown in Table 4. Results show that only all the types of adjacency pairs outperform the Rule-M. Other settings show bad performance on human users test. There are two reason for this: slow convergence and derivation from normal human-human conversation. Above results demonstrate that all the types of adjacency pairs for reward shaping can speed learning and avoid derivation from normal human-human conversation.

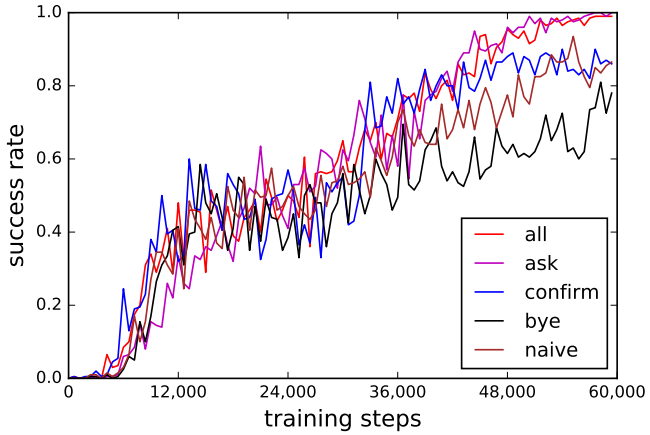

Figure 3. Training curves for different adjacency pairs settings.

Table 4. Human users evaluation on Success Rate (SR) and Average Turns (AT).

	Human Users	
	SR	**AT**
All	0.80	19.80
Ask	0.62	23.22
Confirm	0.30	27.30
Bye	0.22	20.00
Naive	0.32	21.30
Rule-M	0.76	19.56

4.3.2. Comparison of Various Simualtors Settings in One-to-Many Learning

Considering the various simualtors settings in one-to-many learning, we compare the combination of multiple simulators. Since we change the adjacency pairs settings to obtain the different user simulators, we can get 31 combinations based on five seed simulators (i.e., all, ask, confirm, bye and naive). We compare M-MADM-OM with the dialogue managers trained with all combinations containing two simulators, and then show the success rate and average turns in Table 5. We observe that dialogue managers trained with the conbinations containing an all-simulator outperform those dialogue managers trained without the all-simulator on the Group-S and the Rule-S, meanwhile, we observe that all the dialogue managers can achieve the roughly same performance on corresponding trained simulators. We obtain the same results in one-to-three and one-to-four learning. Through the aforementioned results, we think user behaviors generated by the all-simulator can cover user behaviors generated by the Rule-S and the other simulators can generate some user behaviors that the Rule-S can not generate. Thus, we use the combination of five seed simulators to train the M-MADM-OM jointly to improve the robustness and generalization.

Table 5. The different combinations of seed simulators in one-to-two learning on Success Rate (SR) and Average Turns (AT) (Corresponding-S denotes the corresponding training simulators).

	Group-S		**Rule-S**		**Corresponding-S**	
	SR	**AT**	**SR**	**AT**	**SR**	**B**
M-MADM-OM	**0.902**	**18.86**	**0.925**	**17.28**	0.902	18.86
all&ask	0.875	19.42	0.895	19.01	0.905	18.93
all&confirm	0.860	19.64	0.895	18.95	0.910	18.92
all&bye	0.825	20.34	0.905	18.65	0.905	18.85
all&naive	0.835	20.07	0.880	19.27	0.900	18.94
ask&confirm	0.825	24.94	0.645	23.73	0.895	18.91
ask&bye	0.760	21.43	0.645	25.52	0.900	18.89
ask&naive	0.815	20.02	0.550	24.73	0.895	18.93
confirm&bye	0.730	22.08	0.590	18.95	0.895	18.91
confirm&naive	0.725	22.23	0.505	17.04	0.905	18.89

4.3.3. One-to-One Learning vs. One-to-Many Learning

Considering the difference between one-to-one learning strategy and one-to-many learning strategy. The cross-model evaluation is conducted on two dialogue managers: M-MADM-OM and M-MADM-OO, where the M-MADM-OO is optimized via one-to-one learning strategy with all the types of adjacency pairs for reward shaping. For the users in cross-model evaluation, a simulator (MADM-S) trained with M-MADM-OO, Group-S, Rule-S and human users are employed. The results of cross-evaluation on comparing M-MADM-OM with M-MADM-OO is shown in Table 6. Results show that M-MADM-OM outperforms M-MADM-OO in cross-model evaluation, which demonstrates that one-to-many learning strategy can improve the generalization performance of dialogue manager.

Table 6. Cross-model evaluation on Success Rate (SR) and Average Turns (AT).

	MADM-S		**Group-S**		**Rule-S**		**Human Users**	
	SR	**AT**	**SR**	**AT**	**SR**	**AT**	**SR**	**AT**
M-MADM-OM	0.980	17.38	0.902	18.86	0.925	17.28	0.84	18.04
M-MADM-OO	0.975	17.47	0.775	21.27	0.935	18.23	0.78	19.80

5. Conclusions

We introduce a MADM, where an end-to-end dialogue manager cooperates with a user simulator to fulfill a dialogue task. For user simulator reward function, we use the reward shaping technique based on the adjacency pairs to make the simulator learn real user behaviors quickly while learning from scratch. The experimental results show that reward shaping technique speeds up learning and avoids derivation from normal human-human conversation. In addition, we generalize the one-to-one learning strategy to one-to-many learning strategy where a dialogue manager cooperates with various user simulators, which are obtained by changing the adjacency pairs settings. The experimental results also show that the dialogue manager from MADM-OM achieves the best performance on human users involving cross-model evaluation.

In our proposed MADM, there are several models that can be applied to get utterance embedding in dialogue manager, such as TextCNN [38], BERT [39] and XLnet [40]. But these contextualized model is orthogonal to MADM. In the future, we are planning to substitute these models to the bottom bidirectional LSTM in dialogue manager. In addition, we will collect more dataset to enrich the templates expressiveness for NLG and train the models iteratively.

Author Contributions: Methodology, S.L.; formal analysis, X.W.; data curation, S.L.; writing–original draft preparation, S.L.; writing–review and editing, C.Y.; funding acquisition, X.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by NSFC (No.61273365).

Acknowledgments: This paper is supported by 111Project (No. B08004), Beijing Advanced Innovation Center for Imaging Technology, Engineering Research Center of Information Networks of MOE, China. The authors would like to thank the reviewers for their comments and suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

NLU	Natural Language Understanding
DM	Dialogue Management
NLG	Natural Language Generation
DRL	Deep Reinforcement Learning
MADM	Multi-Agent Dialogue Model
HRNN	Hierarchical Recurrent Neural Network
LSTM	Long Short-Term Memory
DNN	Deep Neural Network
MLP	Multi-Layer Perceptron
ST	Success Rate
AT	Average Turns

References

1. Williams, J.D.; Young, S. Scaling POMDPs for spoken dialog management. *TASLP* **2007**, *15*, 2116–2129.
2. Young, S.; Gasic, M.; Thomson, B.; Williams, J.D. POMDP-Based Statistical Spoken Dialog Systems: A Review. *Proc. IEEE* **2013**, *5*, 1160–1179.
3. Gašić, M.; Breslin, C.; Henderson, M.; Kim, D.; Szummer, M.; Thomson, B.; Tsiakoulis, P.; Young, S. On-line policy optimisation of bayesian spoken dialogue systems via human interaction. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 8367–8371.
4. Fatemi, M.; Asri, L.E.; Schulz, H.; He, J.; Suleman, K. Policy networks with two-stage training for dialogue systems. *arXiv* **2016**, arXiv:1606.03152.
5. Su, P.H.; Budzianowski, P.; Ultes, S.; Gasic, M.; Young, S. Sample-efficient Actor-Critic Reinforcement Learning with Supervised Data for Dialogue Management. *arXiv* **2017**, arXiv:1707.00130.
6. Casanueva, I.; Budzianowski, P.; Su, P.H.; Mrkšić, N.; Wen, T.H.; Ultes, S.; Rojas-Barahona, L.; Young, S.; Gašić, M. A benchmarking environment for reinforcement learning based task oriented dialogue management. *arXiv* **2017**, arXiv:1711.11023.
7. Weisz, G.; Budzianowski, P.; Su, P.H.; Gasic, M. Sample Efficient Deep Reinforcement Learning for Dialogue Systems With Large Action Spaces. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2018**, 2083–2097, doi:10.1109/TASLP.2018.2851664.
8. Peng, B.; Li, X.; Gao, J.; Liu, J.; Chen, Y.N.; Wong, K.F. Adversarial advantage actor-critic model for task-completion dialogue policy learning. In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018; pp. 6149–6153.
9. Peng, B.; Li, X.; Gao, J.; Liu, J.; Wong, K.F. Deep Dyna-Q: Integrating Planning for Task-Completion Dialogue Policy Learning. *arXiv* **2018**, arXiv:1801.06176.
10. Liu, B.; Lane, I. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. In Proceedings of the 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), Okinawa, Japan, 16–20 December 2017; pp. 482–489.
11. Ng, A.Y.; Harada, D.; Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In Proceedings of the Sixteenth International Conference on Machine Learning (ICML), Bled, Slovenia, 27–30 June 1999; pp. 278–287.
12. Liddicoat, A.J. Adjacency pairs. In *An Introduction to Conversation Analysis*; Bloomsbury Publishing: London, UK, 2011; pp.143–145.
13. Zhao, T.; Eskenazi, M. Towards End-to-End Learning for Dialog State Tracking and Management using Deep Reinforcement Learning. *arXiv* **2016**, arXiv:1606.02560.

14. Williams, J.D.; Atui, K.A.; Zweig, G. Hybrid Code Networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. *arXiv* **2017**, arXiv:1702.03274.
15. Dhingra, B.; Li, L.; Li, X.; Gao, J.; Chen, Y.N.; Ahmad, F.; Deng, L. Towards End-to-End Reinforcement Learning of Dialogue Agents for Information Access. *arXiv* **2017**, arXiv:1609.00777.
16. Yang, X.; Chen, Y.N.; Hakkani-Tür, D.; Crook, P.; Li, X.; Gao, J.; Deng, L. End-to-end joint learning of natural language understanding and dialogue manager. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; pp. 5690–5694.
17. Pietquin, O.; Geist, M.; Chandramohan, S. Sample efficient on-line learning of optimal dialogue policies with kalman temporal differences. In Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, Barcelona, Spain, 16–22 July 2011; pp. 1878–1883.
18. Scheffler, K.; Young, S. Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning. In Proceedings of the Second International Conference on Human Language Technology Research, San Diego, CA, USA, 24–27 March 2002; pp. 12–19.
19. Cuayáhuitl, H.; Renals, S.; Lemon, O.; Shimodaira, H. Human-computer dialogue simulation using hidden markov models. In Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, San Juan, Puerto Rico, 27 November–1 December 2005; pp. 290–295.
20. Pietquin, O.; Dutoit, T. A probabilistic framework for dialog simulation and optimal strategy learning. *IEEE Trans. Audio Speech Lang. Process.* **2006**, *14*, 589–599.
21. Keizer, S.; Gasic, M.; Mairesse, F.; Thomson, B.; Yu, K.; Young, S. Modelling user behaviour in the HIS-POMDP dialogue manager. In Proceedings of the 2008 IEEE Spoken Language Technology Workshop, Goa, India, 15–19 December 2008; pp. 121–124.
22. Chandramohan, S.; Geis, M.; Lefèvre, F.; Pietquin, O. User Simulation in Dialogue Systems Using Inverse Reinforcement Learning. In Proceedings of the 12th Annual Conference of the International Speech Communication Association, Florence, Italy, 27–31 August, 2011; pp.1025–1028.
23. El Asri, L.; Hem, J.; Suleman, K. A Sequence-to-Sequence Model for User Simulation in Spoken Dialogue Systems. *Interspeech* **2016**, 1151–1155, doi:10.21437/Interspeech.2016-1175.
24. Kreyssig, F.; Casanueva, I.; Budzianowski, P.; Gasic, M. Neural User Simulation for Corpus-based Policy Optimisation of Spoken Dialogue Systems. *arXiv* **2018**, arXiv:1805.06966.
25. Schatzmann, J.; Thomson, B.; Weilhammer, K.; Ye, H.; Young, S. Agenda-based user simulation for bootstrapping a POMDP dialogue system. *NAACL-HLT* **2007**, 149–152, doi:10.3115/1614108.1614146.
26. English, M.S.; Heeman, P.A. Learning mixed initiative dialog strategies by using reinforcement learning on both conversants. *EMNLP* **2005**, 1011–1018, doi:10.3115/1220575.1220702.
27. Chandramohan, S.; Geist, M.; Lefèvre, F.; Pietquin, O. Co-adaptation in spoken dialogue systems. In *Natural Interaction with Robots, Knowbots and Smartphones*; Springer: New York, NY, USA, 2014; pp. 343–353.
28. Das, A.; Kottur, S.; Moura, J.M.; Lee, S.; Batra, D. Learning cooperative visual dialog agents with deep reinforcement learning. In Proceedings of the IEEE International Conference on computer Vision, Venice, Italy, 22–29 October 2017; pp. 2951–2960.
29. Kottur, S.; Moura, J.; Lee, S.; Batra, D. Natural language does not emerge 'naturally' in multi-agent dialog. *arXiv* **2017**, arXiv:1706.08502.
30. Georgila, K.; Nelson, C.; Traum, D. Single-agent vs. multi-agent techniques for concurrent reinforcement learning of negotiation dialogue policies. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, 22–27 June 2014; pp. 500–510.
31. Lewis, M.; Yarats, D.; Dauphin, Y.; Parikh, D.; Batra, D. Deal or No Deal? End-to-End Learning of Negotiation Dialogues. *arXiv* **2017**, arXiv:1706.05125.
32. Bansal, T.; Pachocki, J.; Sidor, S.; Sutskever, I.; Mordatch, I. Emergent complexity via multi-agent competition. *arXiv* **2018**, arXiv:1710.03748.
33. Bernstein, D.S.; Givan, R.; Immerman, N.; Zilberstein, S. The complexity of decentralized control of Markov decision processes. *Math. Oper. Res.* **2002**, *27*, 819-840.
34. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735-1780.
35. Sutton, R.S.; Barto, A.G. Policy gradient methods. In *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 1998; pp.329–331.

36. Schatztnann, J.; Stuttle, M.N.; Weilhammer, K.; Young, S. Effects of the user model on simulation-based learning of dialogue strategies. In Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, San Juan, Puerto Rico, 27 November–1 December 2005; pp. 220–225.
37. Li, X.; Lipton, Z.C.; Dhingra, B.; Li, L.; Gao, J.; Chen, Y.N. A user simulator for task-completion dialogues. *arXiv* **2016**, arXiv:1612.05688.
38. Kim, Y. Convolutional neural networks for sentence classification. *arXiv* **2014**, arXiv:1408.5882.
39. Devlin, J.; Chang, M.W.; Lee, K. Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2019**, arXiv:1810.04805.
40. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.R.; Le, Q.V. Xlnet: Generalized autoregressive pretraining for language understanding. In Proceedings of the 2019 Conference on Neural Information Processing Systems, Vancouver Convention Centre, Vancouver, BC, Canada, 8–14 December 2019; pp. 5754–5764.