The Thirty-First International Florida
Artificial Intelligence Research Society Conference (FLAIRS-31)

A Comparison of Reinforcement Learning Methodologies in Two-Party and Three-Party Negotiation Dialogue

Gang Xiao, Kallirroi Georgila
Institute for Creative Technologies, University of Southern California
12015 Waterfront Drive, Playa Vista, CA 90094, USA

Abstract

We use reinforcement learning to learn dialogue policies in a collaborative furniture layout negotiation task. We employ a variety of methodologies (i.e., learning against a simulated user versus co-learning) and algorithms. Our policies achieve the best solution or a good solution to this problem for a variety of settings and initial conditions, including in the presence of noise (e.g., due to speech recognition or natural language understanding errors). Also, our policies perform well even in situations not observed during training. Policies trained against a simulated user perform well while interacting with policies trained through co-learning, and vice versa. Furthermore, policies trained in a two-party setting are successfully applied to a three-party setting, and vice versa.

Introduction

A dialogue system has a policy which decides on the action that the system should perform given a particular dialogue state (i.e., dialogue context). Reinforcement learning (RL) has become the main approach to automatically learning dialogue policies. In most previous work, RL was applied to slot-filling dialogue domains (e.g., flight reservation).

In this paper, we focus on learning negotiation dialogue policies. Our domain is a collaborative furniture layout negotiation task loosely based on the Design-World task (Walker 1995). Two or more participants have to agree on which furniture items to put in a room. Furniture items and their combinations are worth a specific number of points for each particular negotiator, and this information is only known to that negotiator (until it is revealed in the negotiation). The goal is to earn as a group as many points as possible. The initial conditions (i.e., how many points each furniture item or combinations of items are worth per negotiator) are randomly assigned. Thus the agents need to perform well even for unseen initial conditions, which makes the task very realistic but also very challenging.

We use a variety of RL algorithms, and also vary the number of participants involved in the negotiation. The negotiation dialogue policies are learned against a hand-crafted simulated user (SU) i.e., a model that simulates real user behavior in this task, and/or using co-learning (i.e., two or more agents are trained against one another). Co-learning has the advantage of not requiring a SU for training. For negotiation there is one more reason in favor of co-learning as opposed to learning against a SU. Unlike slot-filling domains, in negotiation the behaviors of the system and the user are symmetric; thus building a good SU is as difficult as building a good system policy. However, co-learning is much harder because each agent is learning against a non-stationary environment, or, in other words, against a moving target (the behavior of the agent or agents it learns against constantly changes because they are also learning at the same time).

We evaluate our policies in a simulation setting (our ultimate goal though is to incorporate the learned policies into a full dialogue system and test them with human users). Our policies achieve the best solution (earn as many points as possible) or a good solution to this negotiation problem for a variety of settings and initial conditions, including in the presence of noise e.g., due to automatic speech recognition (ASR) or natural language understanding (NLU) errors.

Our research contributions are as follows: (1) We compare a variety of RL methodologies (i.e., learning against a SU and co-learning) and algorithms (i.e., Q-learning with and without function approximation, and deep Q-learning) under the same conditions, which has not been done before. (2) For three-party dialogue we combine learning against a SU and co-learning, which has not been done before either. (3) We do not just test our policies against the SU that was used for training them. The training and testing conditions are different, and we learn policies that perform well even in situations not observed during training. Policies trained against a SU perform well while interacting with policies trained through co-learning, and vice versa. Also, policies trained in a two-party setting are successfully applied to a three-party setting, and vice versa. Again, we are not aware of previous work on learning dialogue policies that can be transferred from a co-learning setting to a setting with a SU (and vice versa), or from a two-party setting to a multi-party setting (and vice versa). (4) Unlike most previous approaches to negotiation dialogue policy learning, we simulate various levels of noise to account for ASR and NLU errors.

Related Work

Similarly to us, English and Heeman (2005) learned negotiation policies for a furniture layout negotiation domain based on the Design-World task using co-learning. To make the learning problem more tractable, they worked on a summary state space (an approach that we also use; see below). Heeman (2009) experimented with different representations of the RL state in the same domain (but learning against a handcrafted SU). However, they did not account for ASR or NLU errors, did not compare different methodologies and algorithms, and dealt only with two-party dialogue.

Georgila and Traum (2011) learned argumentation policies against users of different cultural norms in a single-issue negotiation scenario. Then Georgila (2013) learned argumentation policies in a two-issue negotiation scenario. In (Georgila and Traum 2011; Georgila 2013) the policies were designed to work only for specific initial conditions. Papangelis and Georgila (2015) extended the work of Georgila (2013) to four-issue negotiation. They operated on a summary state space, and their learned policies could perform well for a variety of initial settings, including situations that were not observed during training.

Efstathiou and Lemon (2015) used RL to learn negotiation behaviors for a non-cooperative trading game (the Settlers of Catan). The resources of the system were randomly initialized for each dialogue, but it was assumed that the system's adversary always had all the resources that the system would try to trade for. Keizer et al. (2017) compared a variety of trading strategies in the context of the same game. The trading strategies were trained on a corpus of human players playing the game (text-based conversations), and thus did not account for ASR or NLU errors.

Hiraoka et al. (2015) learned trading policies in a multi-party trading scenario, where the dialogue system (learner) trades with up to three other agents. The initial conditions of the traders (except for the learner) were randomly assigned.

Georgila, Nelson, and Traum (2014) used co-learning to learn negotiation policies in a resource allocation scenario. The co-learning problem was cast as a stochastic game, and they focused on comparing single-agent RL versus multi-agent RL (designed specifically for learning against non-stationary environments). They did not allow for a variety of initial conditions. Finally, Lewis et al. (2017) applied end-to-end learning to a multi-issue bargaining negotiation domain. Their models were trained on human-human text data first with supervised learning and then optimized with RL.

Reinforcement Learning

Reinforcement learning (RL) is used for learning the policy of an agent that takes some action to maximize a usually delayed reward. In dialogue, the policy is a mapping function from a dialogue state to a system action. The system's actions are on the speech act level i.e., "inform(item:white table, value:3)" instead of full sentences, to keep the action space tractable. Recently end-to-end approaches to dialogue system building that operate on full sentences have emerged but they are mainly restricted to chatbots due to the difficulty in keeping track of larger dialogue contexts. In task-based dialogue systems the reward function is user satisfaction or task completion. There can be single-agent RL (learning against a stationary environment) and multi-agent RL (learning against a non-stationary environment). Here we employ both single-agent RL in the framework of Markov decision processes (MDPs) and multi-agent RL in the framework of stochastic games (SGs). SGs are a generalization of MDPs for multi-agent RL. In SGs there are multiple interacting agents that select actions, and the next state and rewards depend on the joint action of all the agents. The agents can have different reward functions.

RL requires thousands of interactions between the agent and the environment to learn the optimal policy. In our case, the environment needs to represent the decisions and actions of other negotiators. Thus we use a simulated user (SU) i.e., a model of the user's behavior which will simulate the agent's negotiation partner and help the agent explore the search space of possible policies. The SU can generate a variety of actions for each state based on a probability distribution. We also explore the idea of co-learning where two or more agents can learn at the same time without the need for a SU. For learning against a SU we cast the problem as an MDP (the behavior of the SU does not change as it is not learning), and for co-learning we cast the problem as a SG (the environment of each learning agent is non-stationary).

We experiment with three RL algorithms: simple Q-learning, Q-learning with linear function approximation, and deep Q-learning. In simple Q-learning we need to learn Q-values for all state-action pairs. In Q-learning with linear function approximation the Q-function is a weighted function of state-action features, which allows us to explore commonalities among states. In deep Q-learning we also use function approximation and the weights are the parameters of a neural network. The input to the network are the features and the output the Q-values for each action. We use feed-forward neural networks with two layers and prioritized experience replay. In the future we intend to use algorithms designed specifically for multi-agent RL.

Experimental Setup

In our collaborative negotiation task, each agent has its own preferences initially only known to that agent e.g., the red couch is worth 1 point. As the dialogue progresses, the agents can reveal their preferences making this information public knowledge. The goal is to earn as a group as many points as possible. Apart from simple preferences (e.g., a red couch is worth 2 points), there are also compound preferences (e.g., a blue couch together with a white table are worth 3 extra points). At the beginning of each dialogue, the initial conditions (i.e., simple and compound preferences) are randomly assigned. Thus the agents need to learn to perform well even for unseen initial conditions. In our experiments we assume that there are three furniture items (table, chair, couch) and three colors (blue, white, red).

To make the learning problem more tractable and learn generalizable policies than can be applied to any initial conditions, we work on a *summary state space* rather than the *full state space*. The full state space keeps track of the interaction in detail e.g., the exact proposals that have been made, and the summary state space keeps track of more abstract representations e.g., whether an agent has more information to reveal or not. So in the summary state (i.e., RL state), we keep track of the following variables: whether there is a proposal on the table (true/false), last valid action of the

Preferences of Agent 1
table: blue 0, white 3, red 0
chair: blue 0, white 0, red 1
couch: blue 1, white 0, red 0
red couch + blue table 1
blue couch + blue chair 3
Preferences of Agent 2
table: blue 0, white 1, red 0
chair: blue 0, white 2, red 0
couch: blue 0, white 0, red 3
white couch + blue table 3
white chair + blue table 3
2 best solutions: [blue table, white chair, red couch], [white table, white chair, red couch]
Agent 1: propose(item:blue couch, value:1, expected:4)
Agent 1: inform(item:blue couch, value:1)
Agent 1: inform(items:blue couch,blue chair, value:3)
Agent 2: accept
Agent 2: propose(item:white chair, value:2, expected:5)
Agent 2: inform(item:white chair, value:2)
Agent 2: inform(items:white chair,blue table, value:3)
Agent 1: accept
Agent 1: propose(item:white table, value:3, expected:3)
Agent 1: inform(item:white table, value:3)
Agent 2: confirm{inform(item:blue table, value:3)}
Agent 1: confirm-no, correct{inform(item:white table, value:3)}
Agent 2: accept
Agent 2: replace(item:red couch, expected:3)
Agent 2: inform(item:red couch, value:3)
Agent 1: accept
Accepted: [white table, white chair, red couch]

Figure 1: Example interaction between two agents (each one of them trained through co-learning with 15% noise). "Release turn" actions have been omitted for brevity.

agent (see below), whether the current proposal has been proposed by the agent (true/false), whether the agent has more information to reveal (true/false), whether the agent has a furniture item to replace (true/false), type of current proposal (propose/replace/none), whether there is a furniture type not placed in the room yet (true/false), whether we are in the terminal state of the dialogue (true/false), and whether the current proposal will be accepted by the agent (true/false). Hand-crafted rules are used for moving between the full state and the RL state.

Each agent can perform the following actions: "propose" (if there is a proposal on the table, the new proposal is ignored), "accept" current proposal, "reject" current proposal, "inform" (reveal the agent's preference; an agent may inform multiple times), "replace" a furniture item in the accepted solution (if there is a proposal on the table, the replacement is ignored), and "release turn" and let the other agent or agents perform actions. We also account for ASR and NLU errors, so we have additional actions "confirm", "correct", "confirm-yes", and "confirm-no". Note that the actions of the RL agent are simple "propose", "replace", etc. without any arguments. Then these RL actions are transformed into full actions e.g., "inform(item:blue couch, value:1)" using hand-crafted rules. The agents deal with one furniture item at a time. But they can replace items that have been placed in the room if during the conversation it is revealed that a better solution exists.

For each training configuration (co-learning or against a SU) we perform 2000 iterations. The learning agents are rewarded +5 points when they reach a solution. There are also small rewards when the agent accepts the best item based on its knowledge (+1.5) and for every "inform" action (+0.5; this can be omitted though and still learn good policies), and a small penalty for each agent's action (-0.2). Slightly varying these numbers did not seem to affect the learned policies. We start with very high exploration (almost 100%) and gradually decrease until the policy explores 10% of the time. We also experimented with varying noise i.e., 15% where misunderstandings occur 15% of the time, and 30% where misunderstandings occur 30% of the time. Misunderstandings are simulated by randomly replacing the correct action or action value with a wrong one.

The SU is hand-crafted, deals with one furniture item at a time, and its policy is designed as follows: (1) If the SU has more information to reveal then "inform". (2) If there is no proposal on the table then: (2.1) if there is no accepted solution for a furniture item then "propose"; (2.2) if the SU thinks there is a better item than the one in the solution then "replace"; (2.3) "release turn". (3) If there is a proposal on the table then: (3.1) if this proposal has been proposed by the SU then "release turn"; (3.2) if this proposal has been proposed by other agents then "accept" or "reject" depending on whether the current proposal is the best possible solution based on the SU's knowledge.

An example interaction between two agents is shown in Figure 1. The agents reach one of the two optimal solutions.

Results

Results are shown in Table 1 for Q-learning and SUs only (where all negotiators are SUs), and Table 2 for Q-learning, and for 30% noise which is a quite realistic noise level (trends were similar for 0% and 15% noise). Results for Q-learning with function approximation and deep Q-learning are omitted due to space limitations because they were similar to Q-learning. In all configurations the agents interact for 10000 dialogues. The initial conditions of each dialogue are randomly initialized but it is guaranteed that there will always be one or more optimal solutions. The tables show the percentages of the time that the interacting agents reach the best solution, or a solution (but not the best one). We can also see the average number of total actions per dialogue (including "release turn"). In Table 1 the training and testing conditions are the same e.g., a policy (P1) learned in one configuration (1LA-1SU i.e., 1 learning agent and 1 SU) is tested against a policy (P2) learned in the same configuration (1LA-1SU i.e., 1 learning agent and 1 SU). In Table 2 the training and testing conditions differ e.g., a policy (P1) learned in one configuration (1LA-2SU i.e., 1 learning agent and 2 SUs) is tested against a policy (P2) learned in a different configuration (3LA i.e., 3 learning agents).

P1	P2	Best Solut (%)	Non-best Solut (%)	Avg #Actions
Q-learning				
1LA-1SU	1LA-1SU	85.57	14.43	31.04
2LA	2LA	82.02	17.98	25.86
1LA-2SU	1LA-2SU	69.77	30.23	58.16
2LA-1SU	2LA-1SU	67.69	32.30	53.80
3LA	3LA	67.54	32.37	52.39
SUs only				
2SU	2SU	84.82	15.18	30.96
3SU	3SU	69.55	30.45	57.76

Table 1: Percentages of best solutions and non-best solutions, and average number of agents' actions (including "release turn"; 30% noise). The training and testing conditions of Policy 1 (P1) and Policy 2 (P2) are the same.

P1	P2	Best Solut (%)	Non-best Solut (%)	Avg #Actions
1LA-1SU	1LA-2SU	86.03	13.96	31.09
1LA-1SU	2LA-1SU	83.61	16.39	28.31
1LA-1SU	2LA	83.62	16.38	28.26
1LA-1SU	3LA	83.96	16.04	28.34
1LA-2SU	2LA-1SU	84.71	15.29	28.41
1LA-2SU	2LA	84.31	15.69	28.53
1LA-2SU	3LA	83.96	16.02	28.26
2LA-1SU	2LA	82.81	17.19	25.88
2LA-1SU	3LA	82.57	17.43	25.91
2LA	3LA	82.97	17.03	25.91

Table 2: Percentages of best solutions and non-best solutions, and average number of agents' actions (including "release turn"; 30% noise) for Q-learning. The training and testing conditions of Policy 1 (P1) and Policy 2 (P2) differ.

The agents reach the best solution most of the time, and only very rarely is there no solution. Having only SUs in the negotiation (see Table 1) can serve as a strong baseline for the performance that can be achieved by rational and collaborative agents. As we can see, negotiations that involve learned policies reach similar (or even better) success rates. Success percentages drop a bit for co-learning and for three-party negotiation; both much harder configurations. Table 2 shows that our policies perform well even when they are tested against policies trained in a very different way. Thus policies trained against a SU perform well while interacting with policies trained through co-learning, and vice versa. Also, policies trained in a two-party setting are successfully applied to a three-party setting, and vice versa.

Conclusion

We used RL to learn dialogue policies in a collaborative furniture layout negotiation task. We employed a variety of methodologies (i.e., learning against a SU versus co-learning) and algorithms. Our policies achieved the best solution or a good solution to this problem for a variety of settings and initial conditions, including in the presence of noise (e.g., due to ASR or NLU errors). Our policies performed well even in situations not observed during training. Policies trained against a SU performed well while interacting with policies trained through co-learning, and vice versa. Furthermore, policies trained in a two-party setting were successfully applied to a three-party setting, and vice versa. For future work, we will incorporate these policies into a full dialogue system and test them with human users.

Acknowledgments

This work was supported by the National Science Foundation under grant number IIS-1450656, and the U.S. Army Research Laboratory under contract number W911NF-14-D-0005. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, position, or policy of the National Science Foundation or the U.S. Government, and no official endorsement should be inferred.

References

Efstathiou, I., and Lemon, O. 2015. Learning non-cooperative dialogue policies to beat opponent models: The good, the bad and the ugly. In *Proc. of SemDial*.

English, M. S., and Heeman, P. A. 2005. Learning mixed initiative dialogue strategies by using reinforcement learning on both conversants. In *Proc. of EMNLP*.

Georgila, K., and Traum, D. 2011. Reinforcement learning of argumentation dialogue policies in negotiation. In *Proc. of Interspeech*.

Georgila, K.; Nelson, C.; and Traum, D. 2014. Single-agent vs. multi-agent techniques for concurrent reinforcement learning of negotiation dialogue policies. In *Proc. of ACL*.

Georgila, K. 2013. Reinforcement learning of two-issue negotiation dialogue policies. In *Proc. of SIGDIAL*.

Heeman, P. A. 2009. Representing the reinforcement learning state in a negotiation dialogue. In *Proc. of ASRU*.

Hiraoka, T.; Georgila, K.; Nouri, E.; Traum, D.; and Nakamura, S. 2015. Reinforcement learning in multi-party trading dialog. In *Proc. of SIGDIAL*.

Keizer, S.; Guhe, M.; Cuayáhuitl, H.; Efstathiou, I.; Engelbrecht, K.-P.; Dobre, M.; Lascarides, A.; and Lemon, O. 2017. Evaluating persuasion strategies and deep reinforcement learning methods for negotiation dialogue agents. In *Proc. of EACL*.

Lewis, M.; Yarats, D.; Dauphin, Y. N.; Parikh, D.; and Batra, D. 2017. Deal or no deal? End-to-end learning for negotiation dialogues. In *Proc. of EMNLP*.

Papangelis, A., and Georgila, K. 2015. Reinforcement learning of multi-issue negotiation dialogue policies. In *Proc. of SIGDIAL*.

Walker, M. A. 1995. Testing collaborative strategies by computational simulation: Cognitive and task effects. *Knowledge-Based Systems* 8:105–116.