Transfer Reinforcement Learning Based Negotiating Agent Framework

Transfer Reinforcement Learning Based Negotiating Agent Framework

Siqi Chen[1(✉)], Tianpei Yang[2], Heng You[1], Jianing Zhao[1], Jianye Hao[1], and Gerhard Weiss[3]

[1] College of Intelligence and Computing, Tianjin University, Tianjin 300072, China
siqichen@tju.edu.cn
[2] University of Alberta, Edmonton, Canada
[3] Department of Advanced Computing Sciences, Maastricht University, Maastricht, The Netherlands

Abstract. While achieving tremendous success, there is still a major issue standing out in the domain of automated negotiation: it is inefficient for a negotiating agent to learn a strategy from scratch when being faced with an unknown opponent. Transfer learning can alleviate this problem by utilizing the knowledge of previously learned policies to accelerate the current task learning. This work presents a novel Transfer Learning based Negotiating Agent (TLNAgent) framework that allows a negotiating agent to transfer previous knowledge from source strategies optimized by deep reinforcement learning, to boost its performance in new tasks. TLNAgent comprises three key components: the negotiation module, the adaptation module and the transfer module. To be specific, the negotiation module is responsible for interacting with the other agent during negotiation. The adaptation module measures the helpfulness of each source policy based on a fusion of two selection mechanisms. The transfer module is based on lateral connections between source and target networks and accelerates the agent's training by transferring knowledge from the selected source strategy. Our comprehensive experiments clearly demonstrate that TL is effective in the context of automated negotiation, and TLNAgent outperforms state-of-the-art Automated Negotiating Agents Competition (ANAC) negotiating agents in various domains.

Keywords: Automated negotiation · Transfer learning · Reinforcement learning · Deep learning

1 Introduction

In the domain of automated negotiation, autonomous agents attempt to reach a joint agreement on behalf of human negotiators in a buyer-seller or consumer-provider setup. The biggest driving force behind research into automated negotiation is arguably augmentation of human negotiators' abilities as well as the broad spectrum of potential applications in industrial and commercial domains [2,6]. The interaction framework enforced in automated negotiation lends itself to the use of machine learning techniques for exploring effective

H. Kashima et al. (Eds.): PAKDD 2023, LNAI 13936, pp. 386–397, 2023.
https://doi.org/10.1007/978-3-031-33377-4_30

strategies. Inspired by advances in deep learning (DL) [8,11] and reinforcement learning (RL) [14,15], the application of DRL on negotiation has made significant success [1,3,4,7,9]. However, all these methods need to learn from scratch when faced with new opponents, which is inefficient and impractical.

The existing works mainly focus on how to use the gained experience to train an agent to deal with the encountered opponents [13]. In practice, the agent however may be faced with unfamiliar or unknown opponent strategies, in which its policy may be ineffective, and the agent thus needs to learn a new policy from scratch. Besides, in most negotiation settings, agents are required to negotiate with multiple types of opponents in turn which may be unknown. The problem behind it is that learning in such manner is time-costly and may also restrict its potential performance (e.g., ignoring all previous experience and learned policies that are relevant with the current task). So, a core question arises: how to accelerate the learning process of new opponent strategy, while improving the performance of the learned policy.

This paper describes an attempt to answer the question with transfer learning (TL), which has emerged as a promising technique to accelerate the learning process of the target task by leveraging prior knowledge. We propose a novel TL-based negotiating agent called TLNAgent, which is the first RL-based framework to apply TL in automated negotiation. It comprises three key components: the negotiation module, the adaptation module, and the transfer module. The negotiation module is responsible for interacting with other agents according to the current strategy represented by a deep RL policy and providing information for other modules. The adaptation module measures the helpfulness of the source task concurrently based on the two metrics: similarity between the source opponents and the current opponent, as well as the specific performance of the source policies on the target task. The transfer module is the core of our agent framework, which accelerates the agent's training utilizing the source policies that the adaptation module selects. The comprehensive experiments conducted in the work clearly demonstrate the effectiveness of TLNAgent. Precisely, the performance of TLNAgent is carefully studied from the following aspects:

- The performance of TLNAgent and baselines are compared under standard transfer settings.
- The tournament consisting of recent ANAC winning agents is run to investigate how well TLNAgent performs against state-of-the-art negotiating agents in a broad range of negotiation scenarios.

2 Preliminaries

2.1 Negotiation Settings

The negotiation settings consist of a **negotiation protocol** and a **negotiation environment** [5]. First, the negotiation protocol defines the rules and procedures in the negotiation process. This paper considers the stacked alternating offers protocol, which defines the negotiation as alternating between two

agents who can choose to accept each other's offers or propose new offers in their rounds. The negotiation terminates when both parties agree with an agreement ω, or the allowed negotiation rounds run out. Second, the negotiation environment contains the opponents and domains that the agent interacts with in the negotiation process. The strategy of an opponent makes decision at each round. The negotiation domain is composed of multiple issues and preference profiles of both parties. The preference profiles define the relative importance that an agent assigns to each issue under negotiation, and each agent only knows its own preference profile. The outcome space Ω of the negotiation domain can be denoted by $\Omega = \{\omega_1, \cdots, \omega_n\}$, where ω_i represents different offers available in the i-th domain. The offer ω_i includes an arrangement between two negotiation agents for multiple issues of the domain.

2.2 Reinforcement Learning

Markov Decision Process We model the bilateral negotiation as a MDP represented by a $\langle \mathcal{T}, \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$ tuple. In the negotiation setting of this paper, TLNAgent will be penalized if the negotiation is not finished before the allowed negotiation rounds run out. Therefore, time T which indicates negotiation rounds is an important factor affecting the negotiation. In addition, historical offer is also a key information that affects whether agents accept the last offer or make a new offer. In conclusion, we define the state at time t as

$$\begin{aligned} S_t = \{ & t_r, U_o(\omega_o^{t-2}), U_s(\omega_s^{t-2}), U_o(\omega_o^{t-1}) \\ & , U_s(\omega_s^{t-1}), U_o(\omega_o^t), U_s(\omega_s^t) \} \end{aligned} \tag{1}$$

where $t_r = \frac{t}{T}$ is the relative time denoting the progress of negotiation, and the ω_o and ω_s represent the offers made by the opponent and us at time t, respectively. Since the structures of offers are completely different in diverse environments and the number of offers is spacious, it's difficult to apply the offers directly to MDP modeling. Therefore, we introduce a utility function U to map the specific offer to a value between [0, 1]. This not only contributes to the modeling of the state space but also helps us to define the action space:

$$\begin{aligned} a_t &= u_s^t, \quad u_s^t < 1 \\ U^{-1}(u_s^t) &= \arg\max_{\omega}(U(\omega) - u_s^t), \quad \forall \omega \in \Omega \end{aligned} \tag{2}$$

where U^{-1} is an inverse utility function that maps the utility value to a real offer. The inverse utility function U^{-1} maps the action value given by our agent to an offer ω with the closest utility value in the offer space Ω. The agent receives only one reward during the whole negotiation process based on the negotiation result. If the negotiation results in an agreement ω, the agent receives the final reward corresponding to the utility value $U(\omega)$. Otherwise, if the negotiation fails, both parties receive the same reward -1. The reward function R is defined as follows:

$$R(s_t, a_t, s_{t+1}) = \begin{cases} U_s(\omega_a), & \text{if there is an agreement } \omega_a \\ -1, & \text{if no agreement in the end} \\ 0, & \text{otherwise} \end{cases}$$

3 Transfer Learning Based Agent

3.1 Framework Overview

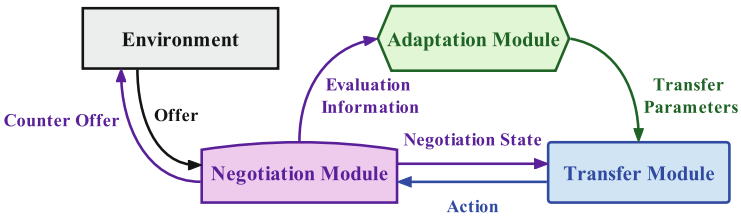


Fig. 1. An overview of our framework

To enable the agent to reuse the learned knowledge and learn how to deal with new opponents, we firstly propose the **Transfer Learning Based Agent For Automated Negotiation** framework (See Fig. 1). The framework is composed of three modules: negotiation module, adaptation module, and transfer module. Through the cooperation of three modules, the framework can accelerate the learning process when encountered a new opponent and improve the learned policy performance. Our framework performs much better than traditional methods based on RL, which will be validated in our experiments.

3.2 Negotiation Module

In this section, we introduce how the negotiation module helps the agent reaches an agreement in a negotiation process. As shown in Fig. 2, the module initializes the session information including the negotiation domain and agent preference in the beginning. Then, the negotiation module generates offers using information sent by transfer module, which implements the bidding policy. Specifically, the negotiation module passes the current state s_t according to Eq. (1) into the transfer module. Subsequently, the negotiation module utilizes Eq. (2) to convert the action a_t given by transfer module to an real offer.

When the agent receives an offer from the opponent, the negotiation module considers two actions: accept or make a counter offer. It first makes a new offer based on the present state. By comparing the utilities which are calculated by utility function $U(\cdot)$ between this offer and the received offer from opponent, the negotiation module decides whether to accept (i.e., accept when the counter offer is better than the new offer), which implements the acceptance policy.

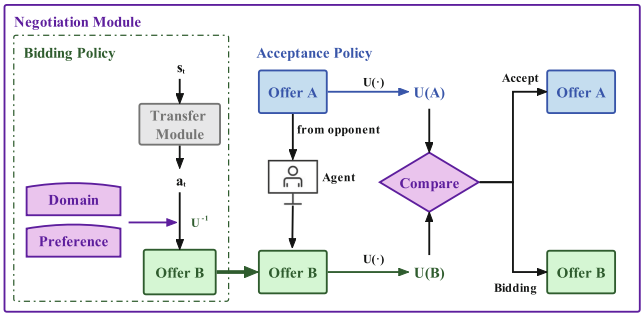


Fig. 2. An illustration of our Negotiation Module which implements the bidding policy and acceptance policy. $U(\cdot)$ and U^{-1} represent the utility function and inverse utility function respectively.

3.3 Adaptation Module

Now we dive into the of details how the adaptation module measures the helpfulness of multiple source policies. In the case of multiple source policies, the primary matter is how to transfer the most relevant knowledge to the target task under different negotiation environments. To solve this problem, we propose two evaluation metrics: performance metric and similarity metric.

As for the performance metric, it is a standard and intuitive approach to directly evaluate the average performance of each source policy when faced with the current opponent. In this work, we use the average utilities $U = \{U_1, \cdots, U_n\}$ of each source policy negotiating with the current opponent in random domains to evaluate them, where $U_n = \frac{1}{I}\sum_{i=1} u_i^n$ and u_i^n denotes the reward value obtained by teacher n for the episode i of evaluation. To ensure the fairness of the negotiation, the evaluation process is only based on the mean results of different domains and is not dependent on the current environment. Subsequently, we pass U through the softmax function to get the weight: $P_{teachers} = \{p_1, \cdots, p_n\}$, where $p_i = \frac{e^{U_i}}{\sum_{i=1}^{n} e^{U_i}}$. The updating of the performance metric is performed continuously throughout the training process and soft changed to ensure the accuracy of the evaluation process and the overall training speed.

The performance metric can obtain the overall performance of source policy when faced with the current opponent. However, it is not rigorous enough to assess the source policy relying on this metric alone because only a part of information in source policies is useful and the performance metric is not fine-grained enough. Therefore, we introduce the Wasserstein distance [10] as our similarity metric to help evaluate the source policy, which compares the similarity between the opponent and the teacher library $O = \{o_1, \cdots, o_n\}$. Specifically, the teacher library contains the opponents used to train source policies. To compare the similarity of the library and the current opponent, we collected our agent's negotiation trajectories τ with different opponents under fixed episodes to calculate the Wasserstein distance. $l_\tau^o = \{H_\tau(\omega_1^o), \cdots, H_\tau(\omega_n^o)\}$ denotes the

probability distribution of offers given by the opponent o in a negotiation trajectory τ, where $H(\cdot)$ is used to calculate the probability of the appearance of the corresponding offer. Then, $l_o = \{\frac{1}{k}\sum_{i=1}^{k} H_{\tau_i}(\omega_1^o), \cdots, \frac{1}{k}\sum_{i=1}^{k} H_{\tau_i}(\omega_n^o)\}$ denotes the average probability distribution of opponent's offers over the k trajectories. Similarly, we can obtain the distribution $L = \{l_1. \cdots, l_n\}$ for different opponents in the teacher library. By comparing the value of $\mathbb{W}$ in Eq. (3), we can get the similarity between the opponent which is used to train source policy and the current opponent.

$$\mathbb{W}(l_i, l_o) = \inf_{\gamma \in \Gamma(l_i, l_o)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|] \tag{3}$$

where $\Gamma(l_i, l_o)$ denotes the set of all joint distributions $\gamma(x, y)$ whose marginals are respectively l_i and l_o. A higher value of $\mathbb{W}(l_i, l_o)$ means that more knowledge in the corresponding source policy will be helpful to the current opponent. Then, the adaptation module takes the inverse of $\mathbb{W}(l_i, l_o)$ and passes it through the softmax function to get the weight $D_{teachers} = \{d_1, \cdots, d_n\}$, where

$$d_i = \frac{\exp(\mathbb{W}(l_i, l_o)^{-1})}{\sum_{i=1}^{n} \exp(\mathbb{W}(l_i, l_o)^{-1})}$$

As our agent's policy is constantly changing in the training process, the similarity metric $D_{teachers}$ will be soft updated every certain number of episodes throughout the negotiation process.

The weighted combination of $P_{teachers}$ and $D_{teachers}$ is used to comprehensively evaluate each source policy. To find the best performance combination, we conducted several experiments to determine the weighting factors μ and λ of the two evaluation metrics described above. The weighted factors are eventually determined as 0.5 and 0.5 for the two approaches based on multiple experiments.

$$W_{teachers} = \mu P_{teachers} + \lambda D_{teachers}$$

In conclusion, the adaptation module measures the helpfulness of each source policy by the two metrics. Then it selects the two most helpful source policies based on $W_{teachers}$ and utilizes their knowledge in the following transfer module.

3.4 Transfer Module

With the guidance of the weighting factors obtained from the adaptation module, the transfer module makes decisions by extracting suitable knowledge from multiple source policies. In the following, we will refer to these source policies as teachers and our agent as student for convenience. Inspired by prior work [12,16], we draw out knowledge directly from teachers' policies and state-value networks using the transfer method of lateral connections. We assume teachers and student have the same number of hidden layers in both the policy and value networks, where N_π and N_V denote the number of hidden layers in the policy networks and state-value networks of teachers and the student respectively. Teacher j's policy networks and state-value networks are represented by $\pi_{\phi'_j}$ and $V_{\psi'_j}$, where the

parameters (ϕ'_j, ψ'_j) are fixed in the training process. In the same, the networks' trainable parameters of the student are represented by (ϕ, ψ).

In the negotiation, the student gets the current state s_t and pass it through teachers' networks to extract the pre-activation outputs of the i-th hidden layers of the j-th teachers' networks:

$$\{h^i_{\phi'_j},\ 1 \le i \le N_\pi,\ 1 \le j \le N\}$$
$$\{h^i_{\psi'_j},\ 1 \le i \le N_V,\ 1 \le j \le N\}$$

To obtain the i-th hidden layer outputs $\{h^i_{\pi_\phi}, h^i_{V_\psi}\}$ of student networks, we performed two weighted linear combinations for the pre-activations of student's networks with the pre-activations of teachers' networks [12,16]:

$$h^i_{\pi_\phi} = p h^i_\phi + (1-p)\sum_{j=1}^{N} w_j h^i_{\phi'_j}$$
$$h^i_{V_\psi} = p h^i_\psi + (1-p)\sum_{j=1}^{N} w_j h^i_{\psi'_j}$$

where $p \in [0, 1]$ is a weighted factor controlling the impact of source policies in the current environment which is increasing with training time. As p increases, source policies have a decreasing influence on our agent in the current environment to avoid the negative transfer. Besides, w_j represents the weight of source policy π_j obtained from the adaptation module. The higher the w_j, the greater the influence of the corresponding π_j on our agent in the current environment, which means the more valuable knowledge and the more helpful for forming our policy. In this way, our agent can leverage the knowledge of multiple source policies to learn a policy to deal with the current opponent.

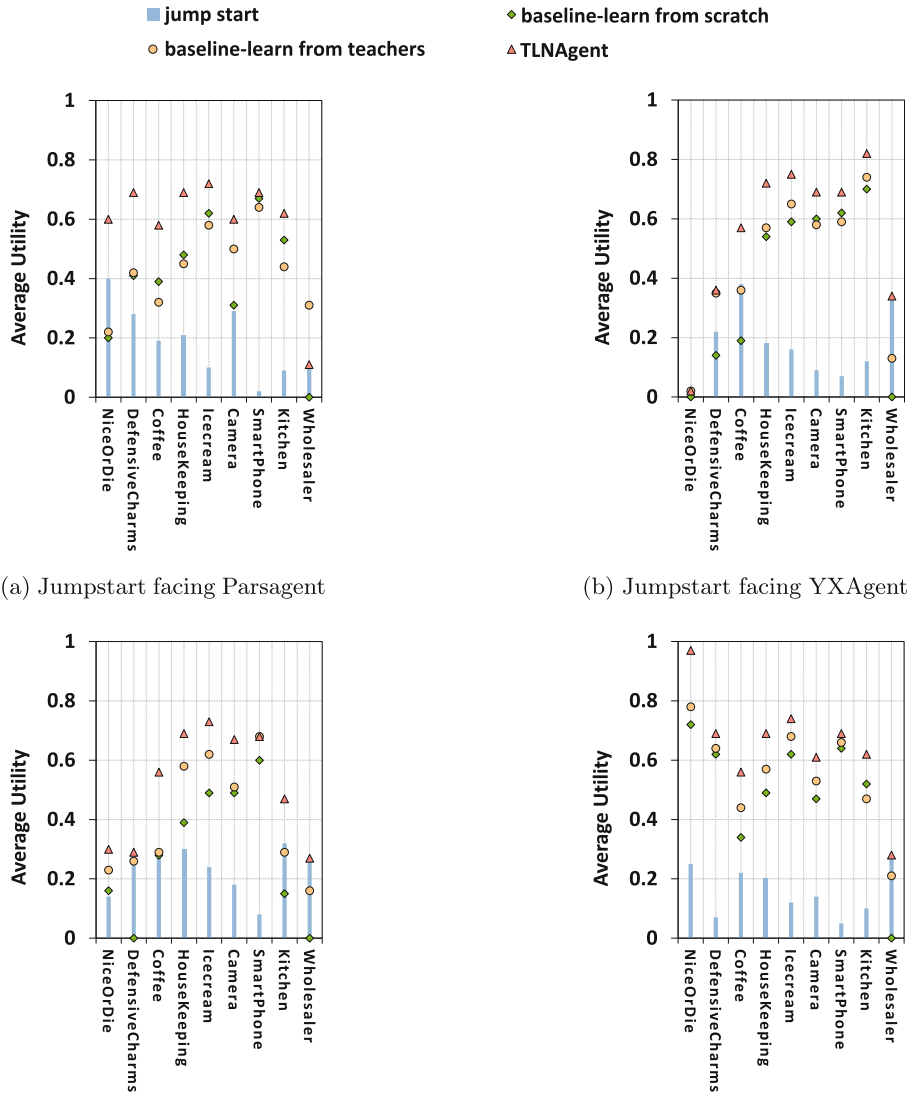
4 Experiments

In this section, we conduct systematic studies to verify the capability of the TLNAgent compared with RL-based methods and other baselines.

Environments: We implemented 11 ANAC winning agents in our negotiation environment to evaluate the negotiation ability of our agent in different scenarios: Atlas3, ParsAgent, Caduceus, YXAgent, Ponpoko, CaduceusDC16, AgreeableAgent2018, Agent36, AlphaBIU, MatrixAlienAgent and TripleAgent. And we used all the 18 domains of ANAC2013 in the experiments.

Baselines: To demonstrate the advantages of using previous knowledge and the superiority of the transfer method when faced with new opponents, we consider the following two baselines in the experiment of Sect. 4.1: 1) Learn from scratch, which uses the standard DRL algorithm SAC and learns without prior knowledge in the new negotiation environment; 2) Learn from teachers, which is directly trained by the opponents that are used to train the source policies.

4.1 New Opponent Learning Task



(a) Jumpstart facing Parsagent

(b) Jumpstart facing YXAgent

(c) Jumpstart facing CaduceusDC16

(d) Jumpstart facing Agent36

Fig. 3. The difference in starting rewards between TLNAgent and other baselines. The dots represent the jumpstarts of different agents. The rectangle represents the difference between TLNAgent and the learning from scratch baseline.

In this section, to verify the efficient learning ability of TLNAgent for previously unknown opponents, we evaluate the agent with multiple tasks consisting

of different opponents and domains. Assume that TLNAgent is only equipped with 4 response policies that are trained by 4 agents in the teacher library as source polices. The teacher library is comprised of Atlas3, Caduceus, Ponpoko and AgreeableAgent2018, which are the champion agents of ANAC from 2015 to 2018. In addition, we consider two baselines (as mentioned above) in the same task for comparison. The opponents of this experiment are Parsagent, YXAgent, CaduceusDC16, and Agent36, which are the second place of ANAC from 2015 to 2018. During the experiment, we train 300,000 rounds for each opponent to ensure our agent and baselines converge, where the allowed number of round per negotiation is 30. The domain used in every training episode is randomly selected among the 18 domains.

The following two transfer metrics are used in experiments, 1) Jumpstart benchmark: the average rewards of TLNAgent and other baselines in the beginning of the task; 2) Transfer ratio: the ratio of mean utility obtained by the agent negotiating with a certain opponent over all 18 domains between TLNAgent and the learn from scratch baseline.

Due to space limitation, we divide all 18 domains into three groups according to their outcome space and select three representative results from each group, as shown in Fig. 3. It can be observed from the results that the jumpstart of TLNAgent is higher than two baselines and has a 50% improvement compared to the baseline learning from scratch. This result indicates that the transfer module can help our agent gain an advantage in the early stage of the negotiation, even if the improvement is not obvious in some scenarios (e.g., the SmartPhone domain).

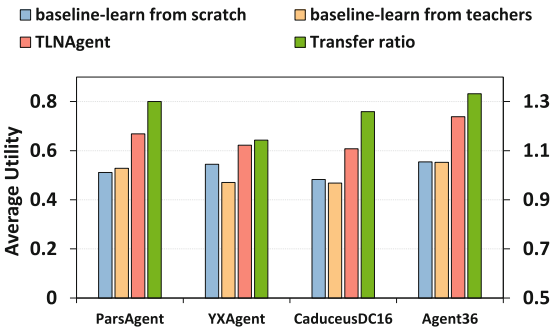


Fig. 4. The average utility of TLNAgent and other baselines when faced with new opponents. The transfer ratio is shown by green bar. (Color figure online)

As shown in Fig. 4, TLNAgent performs better for all opponents, achieving a 26% improvement in average utility compared to the two baselines. This is because TLNAgent transfers helpful knowledge from multiple source policies to the target task learning process through the transfer module. In addition, the adaptation module effectively selects the most appropriate combination of source policies in the current environment so that TLNAgent can decide when and which source policy is more valuable to conduct the adaptive transfer.

4.2 Performance Against ANAC Winning Agents

This section presents the experimental results of a tournament of our agent and 11 ANAC winning agents. To be specific, the experiment consider the top two agents from 2015 to 2018 competitions plus the top three in the 2021 competition[1]. In the tournament, every agent pair will perform a bilateral negotiation of 1000 episodes. The results are shown in Table 1, and the experiments use the following metrics, 1) Average utility benchmark: the mean utility obtained by the agent $p \in A$ when negotiating with every other agent $q \in A$ on all domains D, where A and D denote all the agents and all the domains used in the tournament, respectively; 2) Agreement rate benchmark: the agreement achievement rate between the agent and all others throughout the tournament.

Table 1. Comparison of our proposed TLAgent with 11 ANAC winning agents using average utility benchmark and average agreement achievement rate.

Agent	Average utility	95% confidence interval		Average agreement rate
		Lower Bound	Upper Bound	
Atlas3	0.513	0.487	0.539	0.53
ParsAgent	0.408	0.391	0.425	0.51
Caduceus	0.428	0.415	0.441	0.55
YXAgent	0.474	0.453	0.495	0.37
Ponpoko	0.393	0.382	0.404	0.44
CaduceusDC16	0.452	0.432	0.472	0.53
AgreeableAgent2018	0.533	0.512	0.554	0.79
Agent36	0.315	0.289	0.341	0.47
AlphaBIU	0.572	0.552	0.592	0.64
MatrixAlienAgent	0.558	0.534	0.582	0.59
TripleAgent	0.549	0.532	0.546	0.57
TLAgent	**0.626**	**0.619**	**0.633**	**0.82**

Table 1 shows the performance of our TLNAgent on the average utility benchmark with standard deviation, concurrently with average agreement achievement rate. Our TLNAgent outperforms all ANAC winning agents in the tournament, as exemplified by the higher average utility and higher agreement achievement rate. Without considering the advanced ANAC winning agents of 2021 who have access to past negotiation data, the average utility obtained by our agent is 40% higher than the average benchmark over all other ANAC winning agents. Even when 2021 ANAC winning agents are considered in the comparison, TLNAgent still manages to achieve around 30% advantage in the average utility benchmark. This means that when encountering a new opponent, the agent can utilize the

[1] Note that the themes of ANAC 2019 & 2020 are to elicit preference information from a user during the negotiation, which are different from our negotiation setting.

knowledge of source policies through the adaptation module and transfer module to enhance its negotiation performance rapidly facing the opponent. In addition, TLNAgent achieves the highest agreement rate in the tournament among all agents. The results together show the effectiveness of our framework.

5 Conclusion and Future Work

In this paper we introduced a novel transfer reinforcement learning based negotiating agent framework called TLNAgent for automated negotiation. The framework contains three components: the negotiation module, the adaptation module and the transfer module. Furthermore, the framework adopts the performance metric and the similarity metric to measure the transferbility of the source policies. The experimental results show a clear performance advantage of TLNAgent over state-of-the-art baselines in various aspects. In addition, an analysis was also performed from the transfer perspective.

TLNAgent opens several new research avenues, among which we consider the following as most promising. First, as opponent modeling is another helpful way to improve the efficiency of a negotiation, it's worthwhile investigating how to combine opponent modeling techniques with our framework. Also, it is very interesting to see how well TLNAgent performs against human negotiators. The third important avenue we see is to enlarge the scope of the proposed framework to other negotiation forms.

Acknowledgments. This study was supported by the National Natural Science Foundation of China (Grant No. 61602391).

References

1. Bagga, P., Paoletti, N., Alrayes, B., Stathis, K.: A deep reinforcement learning approach to concurrent bilateral negotiation. In: Proceddings of IJCAI-20 (2020)
2. Chen, S., Ammar, H.B., Tuyls, K., Weiss, G.: Using conditional restricted Boltzmann machine for highly competitive negotiation tasks. In: Proceedings of the 23th International Joint Conference on Artificial Intelligence, pp. 69–75. AAAI Press (2013)
3. Chen, S., Su, R.: An autonomous agent for negotiation with multiple communication channels using parametrized deep Q-network. Math. Biosci. Eng. **19**(8), 7933–7951 (2022). https://doi.org/10.3934/mbe.2022371
4. Chen, S., Sun, Q., Su, R.: An intelligent chatbot for negotiation dialogues. In: Proceedings of IEEE 20th International Conference on Ubiquitous Intelligence and Computing (UIC), pp. 68–73. IEEE (2022)
5. Chen, S., Weiss, G.: An intelligent agent for bilateral negotiation with unknown opponents in continuous-time domains. ACM Trans. Auton. Adapt. Syst. **9**(3), 1–24 (2014). https://doi.org/10.1145/2629577
6. Chen, S., Weiss, G.: An approach to complex agent-based negotiations via effectively modeling unknown opponents. Expert Syst. Appl. **42**(5), 2287–2304 (2015). https://doi.org/10.1016/j.eswa.2014.10.048

7. Chen, S., Yang, Y., Su, R.: Deep reinforcement learning with emergent communication for coalitional negotiation games. Math. Biosci. Eng. **19**(5), 4592–4609 (2022). https://doi.org/10.3934/mbe.2022212
8. Chen, S., Yang, Y., Zhou, H., Sun, Q., Su, R.: DNN-PNN: a parallel deep neural network model to improve anticancer drug sensitivity. Methods **209**, 1–9 (2023). https://doi.org/10.1016/j.ymeth.2022.11.002
9. Lillicrap, T.P., et al.: Continuous control with deep reinforcement learning. In: 4th International Conference on Learning Representations, ICLR 2016, Conference Track Proceedings (2016)
10. Ramdas, A., Trillos, N.G., Cuturi, M.: On Wasserstein two-sample testing and related families of nonparametric tests (2017)
11. Su, R., Yang, H., Wei, L., Chen, S., Zou, Q.: A multi-label learning model for predicting drug-induced pathology in multi-organ based on toxicogenomics data. PLoS Comput. Biol. **18**(9), e1010402 (2022). https://doi.org/10.1371/journal.pcbi.1010402
12. Wan, M., Gangwani, T., Peng, J.: Mutual information based knowledge transfer under state-action dimension mismatch. In: Proceedings of the Thirty-Sixth Conference on Uncertainty in Artificial Intelligence (2020)
13. Wu, L., Chen, S., Gao, X., Zheng, Y., Hao, J.: Detecting and learning against unknown opponents for automated negotiations. In: Pham, D.N., Theeramunkong, T., Governatori, G., Liu, F. (eds.) PRICAI 2021: Trends in Artificial Intelligence (2021)
14. Yang, T., Hao, J., Meng, Z., Zhang, C., Zheng, Y., Zheng, Z.: Towards efficient detection and optimal response against sophisticated opponents. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, pp. 623–629. ijcai.org (2019)
15. Ye, D., et al.: Towards playing full MOBA games with deep reinforcement learning. In: Proceedings of the 34th International Conference on Neural Information Processing Systems (2020)
16. You, H., Yang, T., Zheng, Y., Hao, J., Taylor, M.E.: Cross-domain adaptive transfer reinforcement learning based on state-action correspondence. In: Uncertainty in Artificial Intelligence, Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence (2022)