Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 19(5): 4592–4609.
DOI: 10.3934/mbe.2022212
Received: 19 December 2021
Revised: 17 January 2022
Accepted: 07 February 2022
Published: 07 March 2022

Research article

Deep reinforcement learning with emergent communication for coalitional negotiation games

Siqi Chen, Yang Yang and Ran Su*

College of Intelligence and Computing, Tianjin University, Tianjin, 300072, China

* **Correspondence:** Email: ran.su@tju.edu.cn.

Abstract: For tasks intractable for a single agent, agents must cooperate to accomplish complex goals. A good example is coalitional games, where a group of individuals forms coalitions to produce jointly and share surpluses. In such coalitional negotiation games, how to strategically negotiate to reach agreements on gain allocation is however a key challenge, when the agents are independent and selfish. This work therefore employs deep reinforcement learning (DRL) to build autonomous agent called DALSL that can deal with arbitrary coalitional games without human input. Furthermore, DALSL agent is equipped with the ability to exchange information between them through emergent communication. We have proved that the agent can successfully form a team, distribute the team's benefits fairly, and can effectively use the language channel to exchange specific information, thereby promoting the establishment of small coalition and shortening the negotiation process. The experimental results shows that the DALSL agent obtains higher payoff when negotiating with handcrafted agents and other RL-based agents; moreover, it outperforms other competitors with a larger margin when the language channel is allowed.

Keywords: multi-agent systems; cooperative games; reinforcement learning; deep learning; emergent communication

1. Introduction

The research of single agent learning has made significant progress in various domains, including AlphaGo [1] for Go, Libratus [2] for Texas Hold'em, Watson [3] for medical diagnosis. However, in today's society, more and more tasks require teamwork to complete successfully and effectively, for example, scoring in a football match, search and rescue, and the parliamentary government system of the EU council [4]. Cooperation is thus an indispensable condition for the successful completion of these tasks. On the other side, participants may have different abilities and influence over outcomes, leading to their different interests and a competitive relationship between them when they are selfish

and try to maximize their payoff.

One such example is the weighted voting games (WVGs) — a typical problem in cooperative game theory. It reflects the minimum resources needed to complete a task given each agent has some resources. Agents that are selfish need to form coalitions with other participants and reach agreement on how to share the gains among coalition members, based on the the amount of resources they can contribute. As a powerful mechanism, negotiation is often used to facilitate conflict-resolving and achieve consensus between parties of different interests. This problem can therefore be abstracted into a negotiation problem about team formation, which is the research object of cooperative game theory [5]. Such games are called coalitional negotiation games, and can be naturally modeled by a multi-agent system.

With the development of cooperative game theory, researchers formally quantified actual voting power as the power index. The power index is a function that maps the weighted voting game to the corresponding weight of the players, which reflects the influence of the players on the coalition results [6]. Among them, the Shapley-Shubik power index [7] based on game theory and the Banzhaf index [8] based on probability theory are the earliest influential measurement tools with foundational significance in the field of voting power measurement. Especially the Shapley-Shubik power index, which is a very important and famous kind of power index. The benefit distribution of coalition members based on Shapley value reflects the contribution of coalition members to the coalition, avoids the equalitarianism in distribution, and is more reasonable and fair.

The success of social tasks such as negotiations requires abilities in many aspects [9–11]. During negotiation, the influence of communication [12] on cooperation and negotiation results is the same as or even greater than that of the proposal. Whether the agents can communicate during the negotiation, and how the communication between the agents will affect the negotiation process and results, is what we study in this work. In the weighted voting game, the formation of coalitions and the distribution of benefits require coordination among multiple agents, which provides a good environment for us to study the communication between multiple agents.

In the research of multi-agent cooperation, Foerster et al. firstly introduced communication learning into deep multi-agent reinforcement learning [13]. The reinforced inter-agent learning (RIAL) and differentiable inter-agent learning (DIAL) algorithms they proposed are an extension of the single-agent strategy evolution method deep Q network (DQN) on the multi-agent problem. The DIAL algorithm is an improved version of the RIAL algorithm. Although DIAL can solve the problem of multi-agent collaboration, its algorithm's unidirectional loop structure in communication architecture makes it poor in dealing with rapidly changing environments. CommNet is one of the earliest solutions proposed in the research of multi-agent cooperation problem. Instead of assigning a different neural network to each agent to make decisions, CommNet uses the same network to solve the actions of all agents. However, CommNet can only be used to deal with the same kind of agents, because it takes the form of average value in formula recursion and assumes that all agents have the same weight. Bidirectionally-coordinated nets (BiCNet) combines the advantages of CommNet and DIAL [14], uses Bi-Directional RNN, and uses the deep deterministic policy gradient (DDPG) algorithm as the agent's action strategy. BiCNet is not only able to handle a variety of agent collaboration problems, but also has improved performance in rapidly changing environments. However, it is only a small-scale improvement compared to DIAL.

However, in the research with the background of game, the environment considered by some work

only involves two agents [15, 16]. They regard the problem as a kind of communication rather than the formation of team, thus avoiding the formation of coalition. The work of Matthews et al. proposed a Bayesian reinforcement learning framework to solve the problem of alliance formation [17]. However, the computational cost of Bayesian computation is very high. Bachrach et al. used multi-agent reinforcement learning to study the problem of team formation in negotiation, and it can be applied to any negotiation protocol in a complex environment [18]. However, they did not consider the communication between agents in negotiation. The research results of Crandall et al. show that the use of "cheap talk" can help agents cooperate [19]. However, in their study, cheap talk is carefully designed, and the speech acts of agents are pre-defined. Cao et al. studied the communication between agents on the non-cooperative game model in classic game theory [20]. Their work is based on the offer / counter-offer bargaining games, and according to the concept of cheap talk, they set up a language channel in the game, so that the agent can transmit any character string of arbitrary symbols without a priori basis. However, their research is based on non-cooperative game, and did not investigate whether cheap talk can be generated from the interaction of self-interest agents.

In this paper, deep reinforcement learning (RL) algorithm is used to construct a DALSL (short for Deep Attention LSTM SARSA(λ)) agent to negotiate with others to form coalitions. During the game, the agent negotiates with the payoff obtained in the game as the supervision signal for the success of the task. In addition, we designed specific language channels and language symbols that agents can use to study the influence of communication on agent negotiation. The results show that DALSL agents are better at negotiating than other agents. Moreover, when the game can use the language channel, communication can effectively promote the negotiation, and have a certain impact on the negotiation process and results. DALSL agents can effectively use the language channel, learn better strategies, and reach more favorable agreements in the game.

2. Preliminaries

2.1. Cooperative game

According to Nash's definition of cooperative games, each player can form a small group with other players according to his own interests and cooperate with each other to seek a larger total payment. These small groups are called coalitions, and the coalition composed of all the players is called the grand coalition. Let the set of participants in the game is $N = \{1, 2, ..., n\}$, $|N| = n$ (only considering the limited number of participants), then for any $S \subseteq N$, we call S a coalition of N [21]. When $S = N$, it is called a grand coalition. For a given finite set of participants N, the characteristic function form of the cooperative game is an ordered pair (N, v), where the characteristic function v is the mapping from $2^N = \{S|S \subseteq N\}$ to the set of real numbers $\mathbb{R}$, that is, $(N, v) : 2^N \mapsto \mathbb{R}$, and $v(\phi) = 0$. $v(S)$ is called the characteristic function of coalition S, which represents the benefits that participants in the coalition can obtain from mutual cooperation. Therefore, the characteristic function of the cooperative game means that for each coalition S, $v(S)$ is specified to describe the total amount of transferable utility that the coalition S can obtain without the need for the participants outside of S.

WVG is a typical model of cooperative game. In the WVG, we are given a set of participants $N = \{1, 2, ..., n\}$, each participant $i \in N$ corresponding a weight w_i. When the sum of the weights of the coalition S exceeds the given threshold Q, the members of the coalition can obtain payoff. And the coalition S is called the winning coalition, otherwise, it is called the losing coalition. Therefore, the

characteristic function of the weighted voting game is as follows:

$$v(S) = \begin{cases} 1, & \sum_{i \in S} w_i \geq Q \\ 0, & \sum_{i \in S} w_i < Q \end{cases} \tag{2.1}$$

In a weighted voting game, most of the time, each voter's influence on the voting result does not directly depend on his weight, and it is the influence of each voter on the game result that should be the most important measure of his power. Therefore, we need to find a fair and reasonable quantitative indicator to measure the power of voters. Shapley-Shubik power index proposed by Shapley is a very important kind of power index. It is well known that the order of the voters will have an important impact on the results of the voting. The Shapley-Shubik power index is defined as follows: assuming that the voting is in a certain order, if a voter's approval can make the coalition form a winning coalition, then the probability of the voter's occurrence in the above situation is calculated. The formula for calculating the Shapley value of the ith voter is as follows:

$$\phi_i(N, v) = \frac{1}{|N|!} \sum_{S \subseteq N \setminus \{i\}} |S|!(|N| - |S| - 1)![v(S \cup \{i\}) - v(S)] \tag{2.2}$$

From a mathematical point of view, the Shapley-Shubik power index calculates the average marginal contribution of each voter in an orderly arrangement.

2.2. Multi-agent reinforcement learning

The standard theory of RL is defined by a markov decision process (MDP). By introducing the concept of reward and action into Markov Process, Markov Process can be extended to MDP. In MDP, the immediate reward obtained at the future depends not only on the current state but also on the action that leads to the future state. An MDP can usually be represented by a five-tuple (S, A, P, R, γ), where S represents a set of state space of an environment. A represents the set of actions the agent can choose from. P is a transition probability function $P(s_{t+1}|s_t, a_t)$,specifying the probability that the environment will transition to state $s_{t+1} \in S$ if the agent takes action $a \in A$ in state $s \in S$. R represents the reward function where $r_{t+1} = R(s_t, s_{t+1})$ is a reward received for taking action a_t at state s_t and transfer to the next state s_{t+1}. γ represents the cumulative reward discount coefficient. In MDP, the agent chooses an action a_t according to the policy $\pi(a_t|s_t)$ at state s_t. The environment receives the action, produces a reward r_{t+1} and transfers to the next state s_{t+1} according to the transition probability $P(s_{t+1}|s_t, a_t)$. The objective of RL is to find theoptimal policy π^* for the agent that maximizes the cumulative reward,

However, the agent in MDP needs to have the ability to fully perceive the external environment, that is, the agent needs to obtain global information about the external environment. In the real world, agents usually cannot get the information of the external environment immediately, which makes the agent only make decisions when part of the environment is observable, that is, the agent can only obtain part of the observation information from the external environment, but not the whole observation information. Based on this, partially observable markov decision process (POMDP) is proposed.

POMDP can usually be represented by an octuple $(n, S, A, \Omega, P, O, R, \gamma)$, where n represents the number of agents, S represents the state space, $A = [A_1, A_2 \cdots, A_n$ represents the set of actions of agents. $\Omega = [O_1, O_2 \cdots, O_n$ represents observations available to agents. P is the probability distribution of the agent's alternative actions at a certain moment. O is the observation function, which

Algorithm 1 SARSA

Initialize $Q(s, a)$ arbitrarily
Repeat (for each episode):
  Initialize s
  Choose a from s using policy derived from Q
  Repeat (for each step of episode):
    Take action a, observe r,$s^{'}$
    Choose $a^{'}$ from $s^{'}$ using policy derived from Q
    $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)]$
    $s \leftarrow s^{'}$;$a \leftarrow a^{'}$
  until s is terminal

describes the probability distribution of observation information that the agent may obtain under the given action and result state. $R_i : S \times A_1 \times \cdots \times A_n \rightarrow S'$ represents the reward that the agent can get from the environment after deciding on an action. γ represents the cumulative reward discount coefficient. Markov games also have Markov properties, the next state and reward are only related to the current state and the current joint action. In multi-agent reinforcement learning (MARL), each agent independently learns the optimal behavior strategy $\pi_i : S \rightarrow A_i$ according to its goal – maximizing its own long-term discounted return.

SARSA (State action reward state action) is a traditional RL method, representing the current state s, action a, reward r and the next step s' and a' respectively. We not only need to know the current s, a, r, but also the next step s' and a'. The action decision of SARSA is based on the Q table, which makes decisions based on the Q value in the Q table. The pseudo code of the SARSA algorithm is shown in Algorithm 1, where α is the learning rate. The SARSA(λ) is an improved version of SARSA. The main difference between the two is that after each take action gets the reward, SARSA only updates the previous step $Q(s, a)$, and SARSA(λ) updates the step before the reward. The closer the step to the reward, the more important the step is (the attenuation is controlled by the parameter λ).

2.3. Attentional mechanism

When humans look at the big picture, they usually notice the most attractive part at first glance. The attention mechanism is a bionic of the human visual attention mechanism, and its essence is a resource allocation mechanism [22]. The physiological principle of the attention mechanism is that the human eye only needs to scan the global image quickly to find the target area that needs attention in the image and allocate more attention to the target area, so as to obtain more detailed information and suppress other invalid information.

In recent years, the self-attention mechanism has received a lot of research and application in the field of computer vision [23, 24]. Unlike the standard attention mechanism, it focuses attention on itself, extracts more feature information from the input, and using the "dynamic" attention mechanism to generate different connection weights, so as to capture the relationship itself. This paper uses the self-attention mechanism to improve the efficiency and accuracy of model processing. Please refer to Chapter 4 for details.

3. Coalitional negotiation game

To reflect negotiation dynamics of various cooperative tasks in practice, this work propose a coalitional negotiation game. This game is based on the scenarios of weighted voting game. Following the protocol widely-used in the negotiation domain [25–27], each round of coalitional negotiation game is divided into two stages: proposal and response stage.

In the proposal stage, the player randomly selected by the system first proposes the coalition proposal with the payoff distribution between those potential members. The proposed coalition scheme must include the proposer and the total weight of the coalition members exceeds the threshold Q. In addition to the coalition proposal, players are allowed to send a language message, which we will explain later. Note that all information in the game like weights, exchanged proposals and language messages of all players, the thresholds, weights, and proposer numbers are made public.

The second stage is the response stage. After receiving the proposal, each member of the coalition can choose to agree or disagree according to his consideration on the situation. Only when all coalition members reach agreement, the proposal can be approved, and the total payoff will be distributed according to the payoff distribution scheme, and the game is over. Otherwise, the proposal is rejected, and no payoff is given. The game will continue until any agreement is settled or the maximum round is reached.

Sharing the information and goals between each other through communication is essential to successfully complete the negotiation. In order to study communication on negotiation, two information channels can be used by players. One is the traditional proposal channel, which is used to directly transmit the potential coalition and distribution of payoff. To be exact, the chosen player $i(i \in S)$ proposes a coalition $S(S \in N)$, and $w(S) = \sum_{i \in S} w_i \geq Q$. The payoff distribution scheme $\vec{p}$ satisfies: $\vec{p} \in \mathbb{R}_+^n$, $supp(\vec{p}) = S$, and $\sum_{i \in S} p_i = r$. The benefit r obtained by agent i in the negotiation is related to its own income p_i in the payoff distribution scheme $\vec{p}$, the discount factor γ and the rounds m used to reach the negotiation, $r_i = p_i \times \gamma^m$.

Another one for communicating information is the language channel. Players in the game can use the language channel to transmit character string to communicate in the form of cheap talk [28]. Cheap talk has two key attributes, one is that it is non-binding, and the other is its unverifiable nature, which means that the information sent and received through the language channel will not be enforced and may be a lie. In the negotiation game we specify that the number of symbols that can be used by the agent in the language channel is 5 (characters 0 to 4), and the length of the language message that the agent can generate is 3. Such a setting not only ensures that agents can communicate with enough language symbols, but also avoids the problem that too many characters may make it difficult for agents to learn and use. The process of negotiation between agents is shown in Figure 1.

The experiments are conducted in an online negotiation environment ONECG [29], which supports well the underlying coalitional negotiation games. The visual interface of the ONECG environment is shown in Figure 2. In the negotiation environment, we can easily configure the coalition game specifications, and quickly develop a new agent for negotiation through well-defined APIs. Moreover, when negotiating, agents can not only exchange offers, but also communicate in natural language.

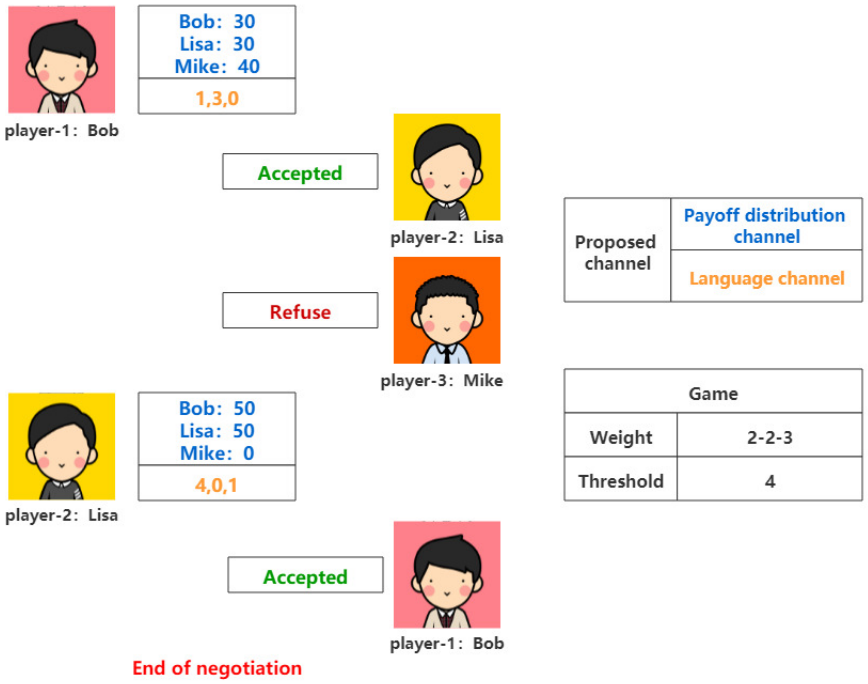


Figure 1. An overview of the negotiation course. In the first stage of the game, an agent starts to make proposals. Its proposal consists of two parts: payoff distribution represented by blue and language information represented by yellow. All players can receive proposals. If a player's income is 0 in the payoff distribution, the proposed coalition of the agent does not include that player. If all the players in the coalition accept the offer, the agreement is reached and the negotiation at this stage ends. Otherwise, negotiations will continue (within the prescribed round of consultations). Note that players who are not included in the offer coalition will only see the offer message and will not have the right to vote.

4. Agent architecture and learning

We propose a DALSL method for constructing coalitional negotiation game agents, and employ MARL method to train it. Each agent uses SARSA(λ) [30] to learn a strategy independently. The neural network — long short term memory network (LSTM) is used to learn Q function in SARSA(λ) to reduce the difficult state space to a manageable number of features and increase the agent's ability to deal with long-term memory, because the negotiation information of the previous round is also very important for the future moves, and deep learning has shown impressive performance in a number of tasks [31–33]. At the same time, we have added attention mechanism for the agent to further improve the negotiating ability of the agent. The weights of the network are optimized by Adam optimizer with default parameters. The algorithm structure is shown in Figure 3.

In the process of interacting with the environment, the SARSA(λ) algorithm will obtain the state s at the current moment, the action a, the reward r, the state $s^{'}$ and the action $a^{'}$ in the next step $t+1$. The model will use a trajectory matrix E to record each step of each round, and store these parameters $e_t = (s, a, r, s^{'}, a^{'})$. As an upgraded version of the SARSA algorithm, the SARSA(λ) algorithm can

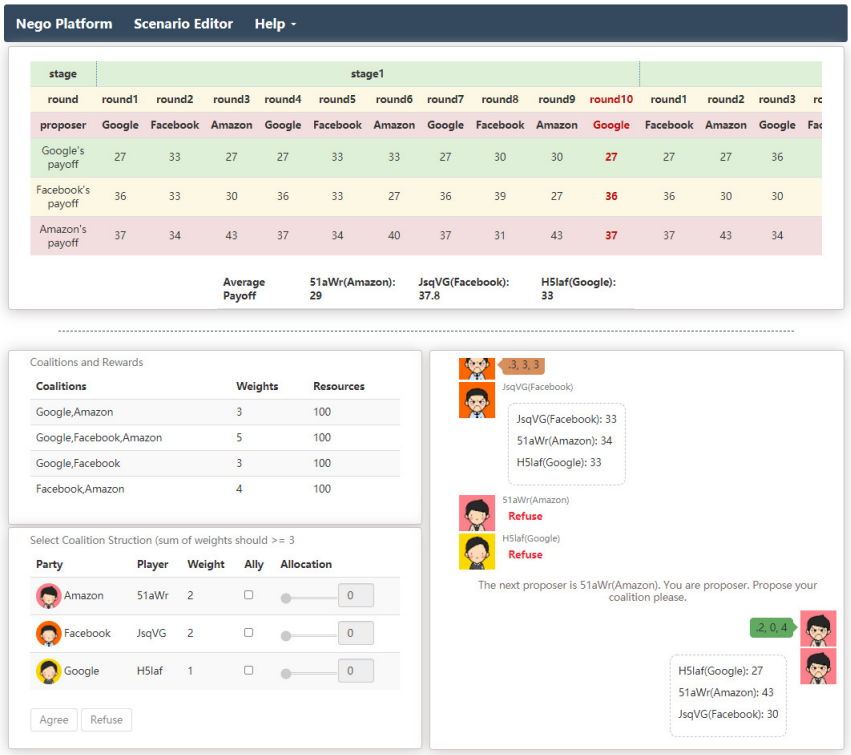
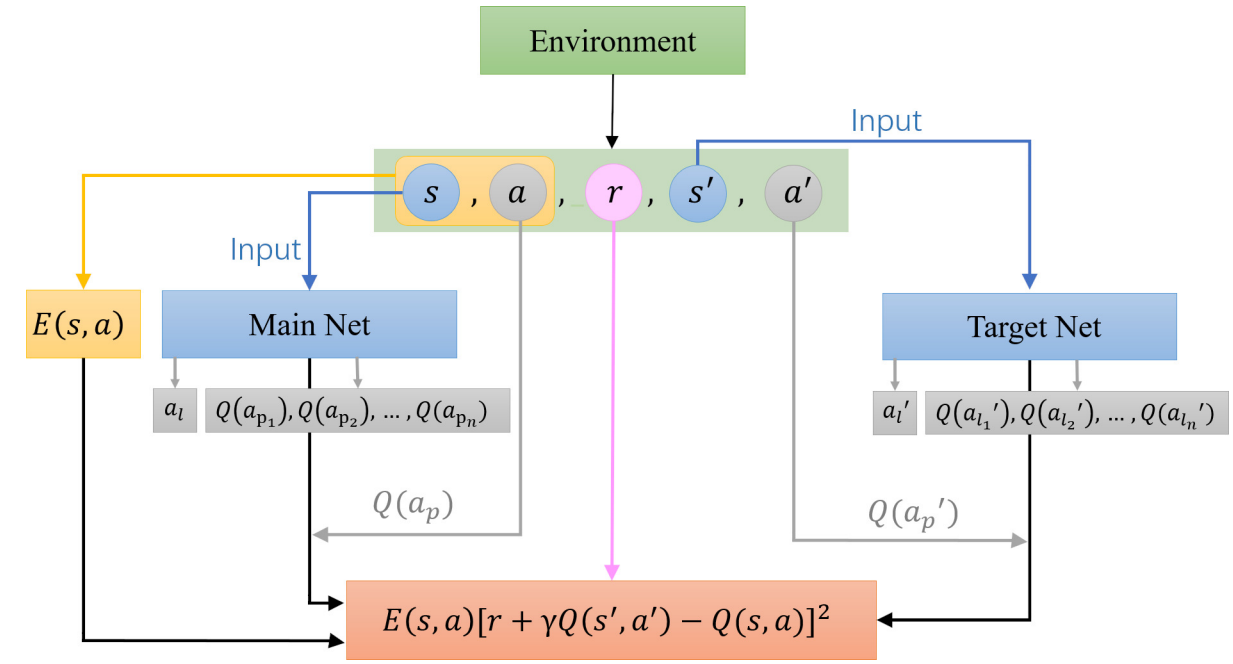


Figure 2. Screenshot of coalitional negotiation dynamics.

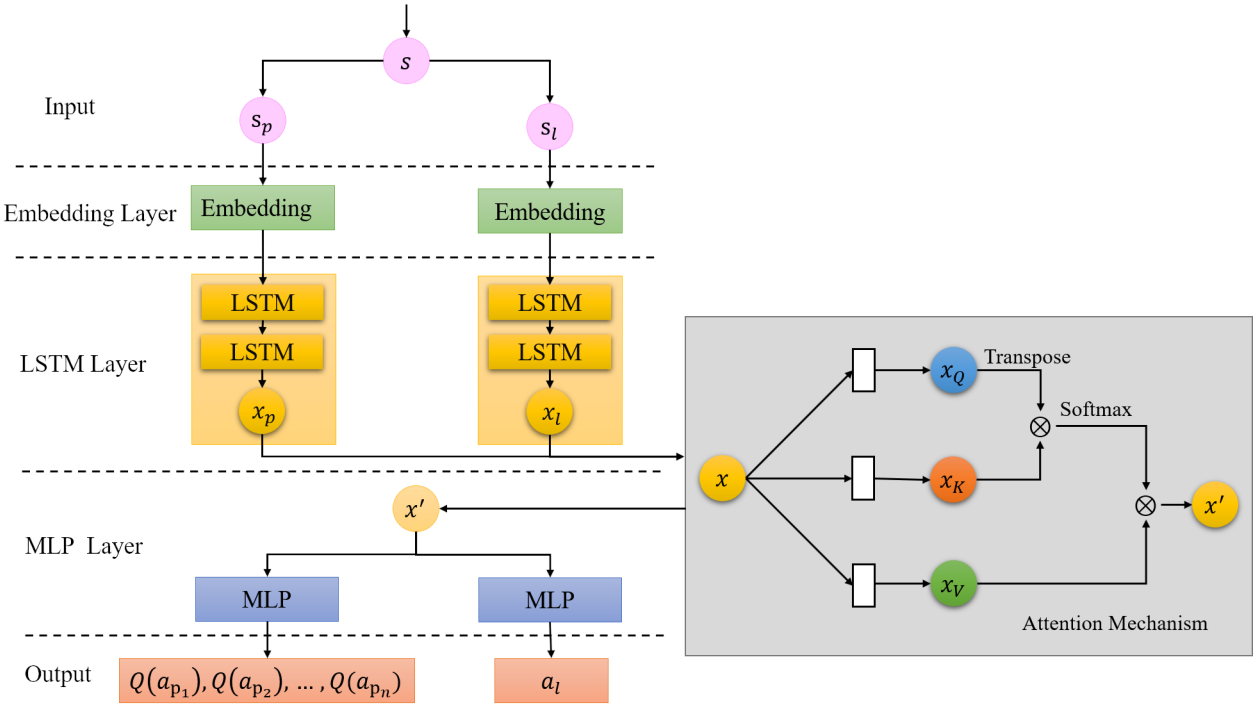
learn how to obtain a higher reward value more efficiently , because the SARSA algorithm only updates the step before the reward value, while the SARSA(λ) algorithm updates the step before the reward value. The value of λ ranges from 0 to 1. If the value of λ is 0, SARSA(λ) is equivalent to SARSA, and only the last step experienced before obtaining the reward value is updated; if the value of λ is 1, the SARSA(λ) algorithm updates all experiences before the reward value is obtained Steps. The trajectory matrix E is used to store the steps taken to obtain rewards, so when the new round starts, E needs to be cleared.

SARSA(λ) uses the ϵ-greedy strategy when choosing actions. The ϵ-greedy strategy is a combination of random and greedy, which balances exploration and utilization. ϵ-greedy strategy specifically refers to the fact that when the agent chooses an action at time t, it does not completely follow the greedy strategy to execute it, but has the probability of ϵ to randomly select the action, and the probability of 1-ϵ selects the action according to the greedy strategy.

The algorithm pseudo code of the DALSL model is shown in Algorithm 2. Two networks are set up in the model, one is the current network Q, the other is the target network $\bar{Q}$, the two networks have the same structure. After the target network is introduced, the target Q value remains unchanged for a period of time, thus reducing the correlation between the current Q value and the target Q value to a certain extent, so as to improve the stability of the algorithm. In SARSA(λ), the update of the Q value relies on the target Q value calculated using the reward value and the Q table. Therefore, the model uses the target Q value as a label and defines the loss function of the network so that the current Q value approaches the target Q value. The loss function is the mean square error of the current network



(a) Overall structure diagram.



(b) The structure of network in Figure (a).

Figure 3. Algorithm structure.

Algorithm 2 DALSL

Initialize replay memory D
Initialize action-value function Q with random weights θ
Initialize target action-value function $\bar{Q}$ with weights $\bar{\theta} = \theta$

For *episode* = 1, M do
  $E(s, a) = 0$, for all $s \in S$, $a \in A(s)$
  Initialize environment state s_1
  For $t = 1$, T do
    Compute LSTM output $\phi_t = \phi(s_t)$
    With probability ξ select a random action a_t
    Otherwise choose a_t from ϕ_t using policy derived from Q
    Execute action a_t and observe reward r_t , state s_{t+1}
    Store transition $(s_t, a_t, r_t, s_{t+1}, a_{t+1})$ in D and E
    Get all transitions $(s, a, r, s^{'}, a^{'})$ from D
    Encode $s, s^{'}$ into $\phi, \phi_{s^{'}}$ with LSTM ϕ
    Set $y = r + \gamma\bar{Q}(\phi_{s^{'}}, a^{'}; \bar{\theta})(1 - d)$
    Set $E(s, a) = \lambda E(s, a)$
    Perform a gradient descent step on $E(s, a)[(y - Q(\phi_s, a; \theta))^2]$ with respect to the network parameters θ
  End for
  Reset D, $\bar{Q} = Q$
End for

Q and the target network $\bar{Q}$:

$$L(\theta) = \mathbb{E}_{s,a,r,s^{'},a^{'}} E(s, a)[Q(s, a|\theta) - y]^2 \tag{4.1}$$

where, $y = r + \gamma\bar{Q}(s^{'}, a^{'}|\bar{\theta})$. θ is the current network parameter, $Q(s, a|\theta)$ represents the output of the current network, used to evaluate the value function of the current state action pair. $\bar{\theta}$is the target network parameter, $\bar{Q}(s^{'}, a^{'}|\bar{\theta})$ represents the output of the target network, which is used to calculate the target Q value. The parameter update of the target network $\bar{Q}(s^{'}, a^{'}|\bar{\theta})$ lags behind θ.

The initial input required for model training is the basic configuration parameters of WVGs, including the weight $\overrightarrow{w} = \{w_1, w_2, ..., w_n\} \in \mathbb{R}^n$ of the agents and the threshold Q needed to reach the coalition. In the model, the action set of the proposed agent consists of two parts. The first part is the payoff distribution action set. The action set of the payoff distribution part is n-tuple $\overrightarrow{p} = \{p_1, p_2, ..., p_n\}$, where $\overrightarrow{p} \in \mathbb{N}_0$ and $\sum_{i \in S} p_i = r$. According to the game setting, the coalition S is composed of agents whose payoff in the payoff distribution is not zero, that is to say, $S = \{i|p_i > 0\}$. The second part is the language message action, which is a sequence of prescribed language symbols (y_1, y_2, y_3), where $y_i \in \{0, 1, 2, 3, 4\}$.

The input s obtained by LSTM at each time step t can be divided into two parts, one is the payoff distribution s_p of the agent, and the other is the language messages s_l of the language channel. First, we use the embedding table to convert the two inputs into dense vectors. Although both inputs are numbers, the meaning of payoff distribution and language messages is not the same, so the two inputs

correspond to two embedding tables respectively. The LSTM is then used to encode each input sequence, and each input sequence still corresponds to an LSTM followed by the attention mechanism, and the result is two fixed-size vectors. Therefore, the output of the LSTM network is the input of the attention mechanism. This article uses the following formula to calculate the attention mechanism:

$$x^{'} = softmax(\frac{x_Q x_K^T}{\sqrt{d_k}})x_V \tag{4.2}$$

As shown in Figure 3(b), the structure of the attention mechanism is divided into three branches from top to bottom, which are the three features x_Q, x_K and x_V obtained after convolution operation. Then transpose the obtained x_Q, use the dot product operation and x_K to calculate the similarity to get the weights, and then use the softmax function to normalize the weights to make the weights of important elements more prominent. The weight and the corresponding key value x_V are dot-producted to get the output of the final attention mechanism $x^{'}$. The factor $\sqrt{d_k}$ in formula (4.2) is the dimension of the column vectors in the input x_Q and x_K, which mainly plays a role of adjustment to avoid the problem of excessive dot product results. If the dot product result is too large, the Softmax function may enter a non-gradient area. Finally, connect the obtained two vectors and pass through the multi-layer perceptron to get the action state value of the agent at the timestep t for the agent to select the action.

Each DALSL agent is independent, and trained to obtain a higher reward value as the only goal. Each agent learns to map the observation results of the environmental state to action decisions. Agents use RL method to update their strategies by interacting with other agents in the environment, and eventually learn an appropriate behavior strategy.

5. Experiment

5.1. Experimental setup

Our experiment was based on a WVG, with three agents participating in each game. In the experiment, the maximum negotiation round of each negotiation is up to 20, and the discount factor is 0.98 (often used in negotiation to reflect the value of time by reduction of the utility over time) [26]. Experiment configurations are based on a Gaussion distribution over underlying weighted voting games. Each sample generated by this distribution is indicated by the weight vector $\overrightarrow{w}$ and the threshold Q. The sampled set are divided them into training set and testing set. In each group of experiments, we train the agent for 100,000 games, and then evaluate the agent on the test set.

Because designing an agent for every coalitional negotiation game is difficult and time-consuming, We have designed two types of three general negotiation agents as benchmarks. One is based on RL types of agents, proposed by Yoram Bachrach et al. [18]. It is the first time that a deep RL agent is used to solve the team formation negotiation problem in cooperative game theory. We implemented it and named it Bachrach agent. This agent focuses on solving the team formation problem in the negotiation and ignores the communication problem in the negotiation. We improves the Bachrach agent so that it can use the language channel and use the specified language symbols in the negotiation. The improved agent is named Bachrach$^+$ agent.

The other two agents are of handcraft type. We use two baselines to design handcrafted agents, one is the weighted-proportional agent and the other is the Shapley-proportional agent. For a given game $(\overrightarrow{w}, Q, r)$, two handcrafted agents randomly select a compliant coalition S. The main difference

Table 1. Average payoff statement of each agent in the negotiation.

	Experiment 1	Experiment 2	Experiment 3	Experiment 4
DALSL agent	31.01	31.57	35.1	34.71
Bachrach$^+$ agent	26.75	–	–	28.39
Shapley-proportional agent	–	25.69	–	26.69
weighted-proportional agent	–	–	20.79	21.81

between the two handcrafted agents is the payoff distribution in the proposed strategy. The payoff distribution of the coalition members of the weight-proportional agent is $p_i = (w_i / \sum_{j \in S} w_j) \times r$ (where w_i is the weight of the agent i), and the payoff distribution of the coalition members of the Shapley-proportional agent is $p_i = (\phi_i / \sum_{j \in S} \phi_j) \times r$ (where ϕ_i is the Shapley value of agent i). In addition to payoff distribution, the proposed strategy also includes language messages, but handcrafted agents cannot freely negotiate with language characters using language channels like RL-based agents, and can only use predefined languages. Therefore, we set the language strategy of the handcrafted agent to a fixed language message. For the acceptance strategy, we compare the payoff offered to the handcrafted agent in the proposal with the payoff p_i that the handcrafted agent would expect to get in the team. $g_i = r_i - p_i$ represent the difference between the quotation received by the handcrafted agent and the payoff it expects to obtain. A positive value of g_i means that the handcrafted agent is getting more payoff than they expected, and a negative value of g_i means that the handcrafted agent is getting less payoff than they expected. The probability of a handcrafted agent accepting an offer is $\sigma(c \cdot g_i)$, where σ represents the logical function $\sigma(x) = \frac{1}{1+e^{-x}}$, where c is a constant that controls the range of g_i between -1 and 1. Therefore, when the quotation obtained by the handcrafted agent is equal to its expected value, the handcrafted agent will have a half probability of accepting the quotation.

5.2. Comparison with other agents

Experimental results will be reported in this section. The first three groups of experiments all used two DALSL agents to negotiate with other types of agents. Experiment 1 is for DALSL agent to negotiate with Bachrach$^+$ agent, Experiment 2 is for DALSL agent to negotiate with Shapley-proportional agent, and Experiment 3 is for DALSL agent to negotiate with weighted-proportional agent. In addition, this section sets up Experiment 4 to conduct an agent random negotiation experiment, that is, randomly select three of the four agents to negotiate to evaluate the negotiation ability of DALSL agents as a whole. The experimental results are shown in Table 1. The average payoff obtained by the DALSL agent is 15.93% higher than the Bachrach$^+$ agent, 22.89% higher than the Shapley-proportional agent, and 68.83% higher than the weighted-proportional agent. We carried out the Mann-Whitney U test on the experimental results, and the test results showed that the difference was significant at the $p < 0.005$ level. It can be seen that the DALSL agent has gained a clear advantage in the negotiation.

The proposal of the weighted-proportional agent is more likely to be accepted, but the payoff is less. In the game where the weight $\overrightarrow{w} = (1, 1, 2)$ and the threshold $Q = 3$, the proposal proposed by the weighted-proportional agent may be $(20, 20, 50)$, $(33, 0, 66)$, $(0, 33, 66)$. The proposal proposed by the Shapley-proportional agent may be $(16, 16, 67)$, $(0, 20, 80)$, $(20, 0, 80)$. It can be found that when the weighting-proportional agent i's weight is $w_i = 2$, its proposal is easier to accept because it "sacrifices" its own payoff. When the weighting-proportional agent i's weight is $w_i = 1$, its proposal

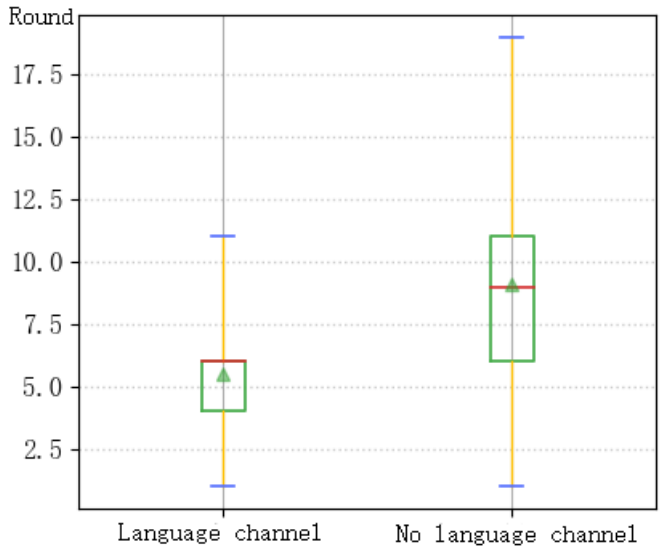


Figure 4. The rounds of agreements reached between RL-based agents.

may be rejected because its own income is slightly higher. The DALSL agent accepts the proposal of the weighted-proportional agent when it "sacrifices" its own payoff, and rejects the proposal when it is not enough. This leads to the high acceptance rate of the weighted-proportion agent's proposal but the low payoff.

In addition, we examine the flexibility of the DALSL agent. We used the weighted-proportional agent (shapley-proportional agent) to train the DALSL agent, but used shapley-proportional agent (weighted-proportional agent) in the evaluation process. This obstacle significantly reduces the average payoff obtained by the DALSL agent in negotiations. However, even in this case, the average payoff obtained by the DALSL agent is still 23.68% higher than the handcrafted agents ($p < 0.005$).

DALSL agent may be sensitive to the initial parameter configuration or the value of hyperparameters during learning. Therefore, we selected robust parameters for the DALSL agent after conducting multiple experiments. Even when the interference learning rate and hidden layer size reach ± 20%, the DALSL agent performs better than handcrafted agents in the same statistical sense.

The experimental results show that the DALSL agent can not only reach a reasonable negotiation agreement, but also superior to some other agents. Although the performance of the DALSL agent may be inferior to more carefully designed handcrafted agents, the DALSL agent has good generalization, and the generated strategies also have a certain degree of flexibility. Moreover, our method provides a way to construct automated and intelligent agents.

5.3. The impact of communication on negotiations

This section uses the DALSL agent to negotiate with the three baseline agents in the case of prohibiting the use of language channels, and contrasts with the content of the previous section to study the influence of communication on the negotiation process and results.

5.3.1. Negotiation between RL-based agents

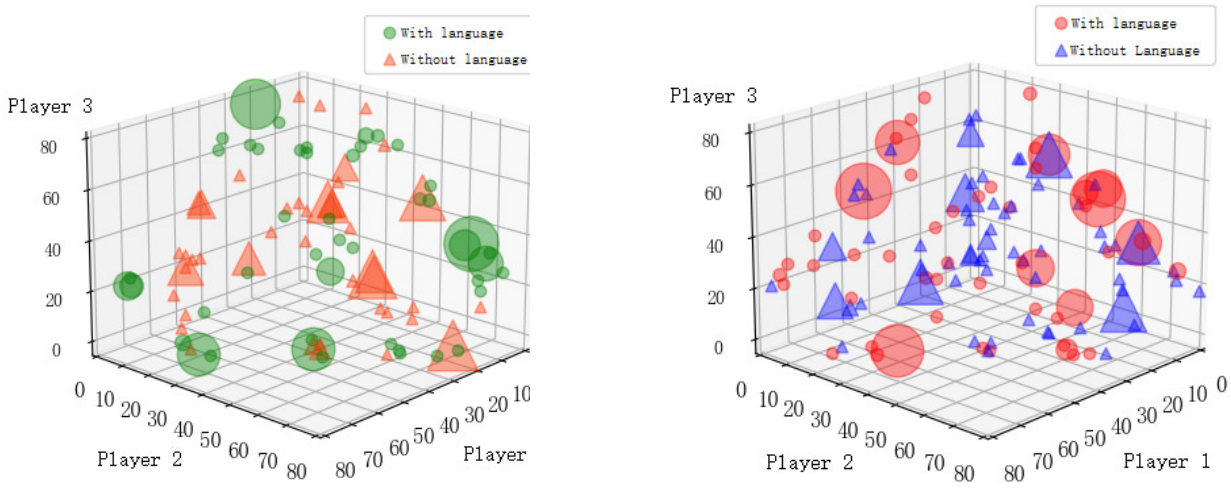
Compared with the experiment that can use the language channel, the payoff obtained by the DALSL agent decreased by 10.55%, and the Bachrach agent decreased by 8.69%. Although the payoff received by the DALSL agent has fallen more, it is still 13.59% higher than the Bachrach agent.

The payoff obtained by the agent is not only related to the agreed payoff distribution, but also related to the discount factor and the rounds used to reach the agreement. As shown in Figure 4, when the language channel can be used, the number of rounds for the agent to reach the negotiation agreement is significantly less.

Continue to analyze and find that the number of rounds is directly related to the coalition reached. In negotiations that can use language channels, the proportion of small coalitions reached as high as 89.50%, while in negotiations that cannot use language channels, the proportion of small alliances reached is only 65.75%. For an individual, since the number of agents distributing payoff in the small coalition is less than that in the ground coalition, there is a greater probability of obtaining higher payoff and also a greater risk of being excluded from the winning coalition.

There are two reasons for the above results. First, the coalition only needs the consent of the members who are included in the coalition,while the agents who are excluded from the coalition do not have the right to vote. Therefore, the achievement of a small coalition requires fewer members to agree. Second, in one stage of the game, agents generally earn more in the small coalition than in the ground coalition because the number of agents in the small coalition is less than the number in the ground coalition.

5.3.2. Negotiation between DALSL agents and handcrafted agent



(a) Shapley-proportional agent

(b) Weight-proportional agent

Figure 5. Scatter plot of payoff distribution reached between DALSL agents and handcrafted agents.

The average payoff obtained by agents in the negotiation of prohibiting the use of language channels, the payoff obtained by DALSL agents decreased by 10.52%, handcrafted agents decreased by 8.15%, while the payoff obtained by DALSL agents was still 39.93% higher than that of handcrafted agents

In the agreement reached, 500 samples were randomly sampled to count the payoff distribution. The results are shown in Figure 5. The reason for random sampling is to avoid too many selected data, which may cause confusion in the image and make it difficult to observe the results. The coordinate axis in Figure 5 represents the payoff obtained by negotiation participants. The circular points indicate that the negotiation is allowed to use language channels, and the triangular points indicate that the negotiation prohibits the use of language channels. The size of the dot shape indicates the number of times the agreement has been reached. Most of the circular points are located on the coordinate axis,which means that in negotiations with language channels, the winning coalitions are more likely to be minor coalitions. This result is consistent with the results obtained from the negotiation of three RL-based agents.

Compared with the negotiation that cannot use the language channel, in the negotiation that can use the language channel, the probability that the winning coalition is a small coalition composed of two DALSL agents has increased significantly by 12.36%. In contrast, the probability that the winning coalition is a small coalition composed of handcrafted agents and DALSL agents dropped by 10.17%. Due to the use of language channels in negotiations, small coalitions are easier to be formed, and at the same time, two handcrafted agents are more likely to be excluded from the winning coalition by the DALSL agent. This also leads to less payoff for handcrafted agents.

Even in a game where only two DALSL agents can use the language channel, the DALSL agent can still use the language channel to form a more favorable small coalition. At the same time, language channel makes DALSL agents form a more stable cooperative relationship, and promotes the cooperation between DALSL agents.

6. Conclusions

The contributions of this paper are three-fold: First, we propose a autonomous negotiation agent — DALSL based on deep RL to perform formation task, which can outperform some handcrafted agents and RL-based agents. Secondly, DALSL agent is completely empirically driven and can be well generalized without the need for manual adjustment and manual data. Finally, the use of language channels enables DALSL agents to achieve better performance in negotiations. In the negotiation, the use of language channels enables RL-based agents to exchange certain information, promote the achievement of small coalitions, and shorten the negotiation process. Moreover, communication promotes the cooperation between the DALSL agents, and makes DALSL agent form a more stable cooperative relationship.

The success of DALSL agent opens several new research avenues, among which we consider the following as most promising. First, it is of interest to study how team formation affects the emergent language used during negotiation process. Then, in addition to agent-agent negotiation, human-agent negotiation is also an important form and has a wide range of applications in our lives. Therefore extending current work to build a reinforcement learning agent capable of dealing with human partners using natural language is believed to further improve the value of the proposed method.

Acknowledgments

This work is supported by National Natural Science Foundation of China (Grant Number: 61602391).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, et al., Mastering the game of go with deep neural networks and tree search, *Nature*, **529** (2016), 484–489. https://doi.org/10.1038/nature16961
2. M. Moravcík, M. Schmid, N. Burch, V. Lisý, D. Morrill, N. Bard, et al., Deepstack: Expert-level artificial intelligence in no-limit poker, *CoRR*, abs/1701.01724.
3. M. V. Devarakonda, C. Tsou, Automated problem list generation from electronic medical records in IBM watson, in *Proc. Twenty-Ninth AAAI Conf. Artif. Intell.*, (eds. B. Bonet, S. Koenig), (2015), 3942–3947.
4. D. Leech, Designing the voting system for the council of the european union, *Public Choice*, **113** (2002), 473–464. https://doi.org/10.1023/A:1020877015060
5. O. Shehory, S. Kraus, Methods for task allocation via agent coalition formation, *Artif. Intell.*, **101** (1998), 165–200. https://doi.org/10.1016/S0004-3702(98)00045-9
6. Y. Zick, K. Gal, Y. Bachrach, M. Mash, How to form winning coalitions in mixed human-computer settings, in *Proc. Twenty-Sixth Int. Joint Conf. Artif. Intell., IJCAI* (ed. C. Sierra), (2017), 465–471. https://doi.org/10.24963/ijcai.2017/66
7. L. S. Shapley, M. Shubik, A method for evaluating the distribution of power in a committee system, *Am. political Sci. Rev.*, **48** (1954), 787–792. https://doi.org/10.2307/1951053
8. J. F. Banzhaf III, Weighted voting doesn't work: A mathematical analysis, *Rutgers L. Rev.*, **19** (1964), 317.
9. L. Wu, S. Chen, X. Gao, Y. Zheng, J. Hao, Detecting and learning against unknown opponents for automated negotiations, in *PRICAI 2021: Trends in Artificial Intelligence* (eds. D. N. Pham, T. Theeramunkong, G. Governatori and F. Liu), (2021), 17–31. https://doi.org/10.1007/978-3-030-89370-5_2
10. X. Gao, S. Chen, Y. Zheng, J. Hao, A deep reinforcement learning-based agent for negotiation with multiple communication channels, in *2021 IEEE 33nd Int. Conf. Tools with Artif. Intell. (ICTAI)*, (2021), 868–872. https://doi.org/10.1109/ICTAI52525.2021.00139
11. C. Gao, J. Liu, Network-based modeling for characterizing human collective behaviors during extreme events, *IEEE Trans. Syst. Man Cybern. Syst.*, **47** (2017), 171–183. https://doi.org/10.1109/TSMC.2016.2608658

12. H. Mao, Z. Zhang, Z. Xiao, Z. Gong, Y. Ni, Learning multi-agent communication with double attentional deep reinforcement learning, *Auton. Agents Multi Agent Syst.*, **34** (2020), 32. https://doi.org/10.1007/s10458-020-09455-w

13. J. N. Foerster, Y. M. Assael, N. De Freitas, S. Whiteson, Learning to communicate with deep multi-agent reinforcement learning, *arXiv prints*, arXiv:1605.06676.

14. P. Peng, Y. Wen, Y. Yang, Q. Yuan, Z. Tang, H. Long, et al., Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games, *arXiv prints*, arXiv:1703.10069.

15. T. Eccles, Y. Bachrach, G. Lever, A. Lazaridou, T. Graepel, Biases for emergent communication in multi-agent reinforcement learning, in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, Vancouver, BC, Canada* (eds. H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, R. Garnett), (2019), 13111–13121. https://dl.acm.org/doi/10.5555/3454287.3455463

16. E. Hughes, T. W. Anthony, T. Eccles, J. Z. Leibo, D. Balduzzi, Y. Bachrach, Learning to resolve alliance dilemmas in many-player zero-sum games, in *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20* , (eds. A. E. F. Seghrouchni, G. Sukthankar, B. An, N. Yorke-Smith), (2020), 538–547.

17. T. Matthews, S. Ramchurn, G. Chalkiadakis, Competing with humans at fantasy football: Team formation in large partially-observable domains, in *Proc. AAAI Conf. Artif. Intell.*, **26** (2012). https://ojs.aaai.org/index.php/AAAI/article/view/8259

18. Y. Bachrach, R. Everett, E. Hughes, A. Lazaridou, J. Z. Leibo, M. Lanctot, et al., Negotiating team formation using deep reinforcement learning, *Artif. Intell.*, **288** (2020), 103356. https://doi.org/10.1016/j.artint.2020.103356

19. J. W. Crandall, M. Oudah, F. Ishowo-Oloko, S. Abdallah, J. F. Bonnefon, M. Cebrian, et al., Cooperating with machines, *Nat. Commun.*, **9** (2018), 1–12. https://doi.org/10.1038/s41467-017-02597-8

20. K. Cao, A. Lazaridou, M. Lanctot, J. Z. Leibo, K. Tuyls, S. Clark, Emergent communication through negotiation, in *6th Int. Conf. Learn. Represent., ICLR 2018, Vancouver, Conference Track Proceedings*, 2018.

21. Y. Shoham, K. Leyton-Brown, *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*, Cambridge University Press, 2008. https://dl.acm.org/doi/abs/10.1145/1753171.1753181

22. D. K. Kim, S. Omidshafiei, J. Pazis, J. P. How, Crossmodal attentive skill learner: learning in atari and beyond with audio—video inputs, *Auton. Agent. Multi Agent Syst.*, **34** (2020), 1–21. https://doi.org/10.1007/s10458-019-09439-5

23. R. Su, Y. Zhu, Q. Zou, L. Wei, Distant metastasis identification based on optimized graph representation of gene interaction patterns, *Brief. Bioinform.*, **23**, (2022), bbab468. http://doi.org/10.1093/bib/bbab468

24. J. Liu, R. Su, J. Zhang, L. Wei, Classification and gene selection of triple-negative breast cancer subtype embedding gene connectivity matrix in deep neural network, *Brief. Bioinform.*, **22**, (2021), bbaa395. https://doi.org/10.1093/bib/bbaa395

25. A. Rubinstein, Perfect equilibrium in a bargaining model, *Econometrica*, **50** (1982), 97–109.

26. S. Chen, G. Weiss, An intelligent agent for bilateral negotiation with unknown opponents in continuous-time domains, *ACM Trans. Auton. Adapt. Syst.*, **9** (2014), 16:1–16:24. https://dl.acm.org/doi/10.1145/2629577

27. S. Chen, G. Weiss, An approach to complex agent-based negotiations via effectively modeling unknown opponents, *Expert Syst. Appl.*, **42** (2015), 2287–2304. https://doi.org/10.1016/j.eswa.2014.10.048

28. K. Dautenhahn, Socially intelligent robots: dimensions of human–robot interaction, *Philos. Trans. R. Soc. B Biol. Sci.*, **362** (2007), 679–704. https://doi.org/10.1098/rstb.2006.2004

29. S. Chen, Y. Cui, C. Shang, J. Hao, G. Weiss, Onecg: Online negotiation environment for coalitional games, in *Proc. 18th Int. Conf. Auton. Agent. MultiAg. Syst.*, (2019), 2348–2350. https://dl.acm.org/doi/10.5555/3306127.3332108

30. G. A. Rummery, M. Niranjan, On-line Q-learning using connectionist systems, University of Cambridge, Department of Engineering Cambridge, UK, 1994.

31. R. Su, X. Liu, L. Wei, Q. Zou, Deep-Resp-Forest: A deep forest model to predict anti-cancer drug response, *Methods*, **166** (2019) 91–102. https://doi.org/10.1016/j.ymeth.2019.02.009

32. R. Su, X. Liu, Q. Jin, X. Liu, L. Wei, Identification of glioblastoma molecular subtype and prognosis based on deep mri features, *Knowl. Based Syst.*, **232** (2021), 107490. https://doi.org/10.1016/j.knosys.2021.107490

33. R. Su, H. Wu, B. Xu, X. Liu, L. Wei, Developing a multi-dose computational model for drug-induced hepatotoxicity prediction based on toxicogenomics data, *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **16** (2018), 1231–1239. https://doi.org/10.1109/TCBB.2018.2858756

AIMS Press