Ann Math Artif Intell
DOI 10.1007/s10472-015-9450-1

A unifying learning framework for building artificial game-playing agents

Wenlin Chen · Yixin Chen · David K. Levine

Abstract This paper investigates learning-based agents that are capable of mimicking human behavior in game playing, a central task in computational economics. Although computational economists have developed various game-playing agents, well-established machine learning methods such as graphical models have not been applied before. Leveraging probabilistic graphical models, this paper presents a novel sequential Bayesian network (SBN) framework for building artificial game-playing agents. We show that many existing agents, including reinforcement learning, fictitious play, and many of their variants, have a unified Bayesian explanation within the proposed SBN framework. Moreover, we discover that SBN can handle various important settings of game playing, allowing for a broad scope of its use in economics. SBN not only provides a unifying and satisfying framework to explain existing learning approaches in virtual economies, but also enables the development of new algorithms that are stronger or have fewer restrictions. In this paper, we derive a new algorithm, Hidden Markovian Play (HMP), from the generic SBN model to handle an important but difficult setting in which a player cannot observe the opponent's strategy and payoff. It leverages Markovian learning to infer unobservable information, leading to higher quality of the agents. Experiments on real-world field experiments in evaluating economies show that our HMP model outperforms the baseline algorithms for building artificial agents.

Keywords Behavioral game theory · Graphical model · Multi-agent learning · Fictitious play · Hidden Markov model

W. Chen (✉) · Y. Chen · D. K. Levine
Washington University in St. Louis One Brookings Drive, St. Louis, MO 63130 USA
e-mail: wenlinchen@wustl.edu

Y. Chen
e-mail: chen@cse.wustl.edu

D. K. Levine
e-mail: david@dklevine.com

Published online: 31 January 2015

1 Introduction

An important need for developing better economic policy prescriptions is an improved method of validating theories. Originally, economics depended on field data from surveys. Laboratory experiments have added a new dimension: a good theory ought to be able to predict outcomes in the artificial world of the laboratory. Modern economists have extended this in two directions: to field experiments and Internet experiments. These innovations are important, as they are easier, faster, and more practical to validate theories [35]. On the other hand, all types of experiments have important limitations. The largest Internet experiment is orders of magnitudes smaller than a small real economy: thousands of subjects rather than millions of real decision-makers. Experiments are still time-consuming and subjects are expensive to pay for large experiments. Finally, control in experiments is still and necessarily imperfect. In particular, it is not practical to control for either risk aversion or social preferences.

An alternative method of validating theories is through the use of artificial or virtual agents. If a virtual world is an adequate description of a real economy, then a good economic theory ought to be able to predict outcomes in that setting. An artificial environment offers enormous advantages over the laboratory, field, and Internet, as it provides complete control and great speed in creating economies and validating theories. In physical sciences, the large computer models used in testing nuclear weapons are an analogy.

The state of the art in the study of virtual economies is agent-based modeling where human behaviors are simulated by artificial gaming agents. Agent-based modeling methods for building virtual economies have lots of benefits and are used in many real-world applications, including organizational simulation and market forecast [4]. The validity of using artificial game-theoretic agents has been validated through numerous experiments [23, 25]. Hence, it is vital for economics study to improve the quality and versatility of the artificial agents, produced by algorithms that learn from the playing history of human players.

In agent-based virtual economies, each artificial gaming agent plays its role and interacts with other agents, from which we could witness and investigate human behavior. Game theory is the study of strategic iteration among intelligent players, and widely used in economical and industrial areas. However, game theory has traditionally been developed as a theory of handling situations where players exhibit equilibrium behaviors [24]. For economy or other real-world studies, it is not enough to analyze the equilibriums and it is important to learn from real human behaviors. It is long known that human behaviors often deviate from the Nash equilibrium, due to reasons such as information constraints, lack of total rationality, and the potential of human collaboration in repeated play. With these limitations, there even does not exist equilibrium in some cases [6]. Thus, it is important to develop models to describe the game-playing process rather than just analyzing the equilibrium, by learning from historical data of human subjects playing games.

Two major classes of such machine learning models are reinforcement learning[1] [24] and fictitious play [27]. However, they are rather simple rule-based models that are not good representation of the sophisticated behaviors of human. Each of these models assumes certain information that is often unavailable. Moreover, they cannot make use of certain extra information to achieve better modeling quality even if it is available.

[1]The term "reinforcement learning" has broad meanings in various learning contexts. In this paper, we use this term to refer to the specific technique for building game-playing agents.

The diversity of human rationality and game-playing settings lead to different game-playing processes. It is desirable to have a unifying framework, under which a broad class of models capable of mimicking human behaviors under different settings are described. In this paper, we explore the application of probabilistic graphical models in building such agents. We propose a general sequential Bayesian network (SBN) model to capture behavioral incentive structure of human players under different game-playing settings. Our work shows that it is promising to apply established machine learning methods to agent-based economics. This paper makes the following contributions:

- We propose a unifying SBN framework that can handle a broad class of game-theoretic settings. SBN works across different settings and is able to take full advantage of all the available data under these settings. In essence, we use statistical inference to derive beliefs on unobservable variables based on observed data.
- Based on the SBN framework, we elucidate the close relationship between reinforcement learning and fictitious play, two learning algorithms that have been developed in separation. We discover that both algorithms as well as their variants have a unifying Bayesian explanation.
- Based on the SBN framework, we further derive a new learning method, called Hidden Markovian Play (HMP), which leverages Markovian learning to infer unavailable information from observed data, leading to higher modeling quality of the artificial agents. Moreover, the new algorithm requires less observable information than some existing models.
- We apply the proposed HMP model to some real-world field experiments in evaluating economies such as market entries. The results show that the HMP model outperforms many existing algorithms for building artificial agents.

This paper is organized as follows. We formally define the problem of learning human behaviors in game-playing process and the methodology for evaluating performance in Section 2. Then, we survey the related work in Section 3. We describe our SBN framework in Section 4 and explain how it encompasses previous learning approaches in Section 5. Based on the SBN framework, we propose the HMP algorithm, a novel game-playing agent in Section 6. We evaluate the performance of HMP, along with other existing agents, on data from real-world field experiments in Section 7. Finally, we draw conclusions in Section 8.

2 Learning human behaviors in playing games

In this paper, we focus on repeated games which are the main type of games describing long-term economical phenomena. Our task is to build artificial agents, models that predict human behavior statistics given a game-theoretic environment. It can be viewed as a machine learning problem: we build an artificial model by learning from the existing data of human players participating in games.

In a repeated game, players play a same game repeatedly. In each round, players simultaneously submit their strategies and receive payoff determined by the utility (payoff) function. Consider a n-person normal form game where players are indexed by $i, i = 1, 2, ..., k$. The strategy space of player i contains m^i discrete strategies, i.e.

$$S^{(i)} = \left\{ s_1^{(i)}, s_2^{(i)}, ..., s_{m^i}^{(i)} \right\}.$$

$S = S^{(1)} \times ... \times S^{(n)}$ is the strategy profile based on which individual payoff is assigned in each round. We use s^i to denote a strategy variable of player i and s^{-i} to denote the strategy combination of all players except i, i.e.

$$s^{-i} = (s^1, ..., s^{i-1}, s^{i+1}, ..., s^n).$$

The utility function of player i is $u^i(s^i, s^{-i})$, which maps the strategy profile to payoff. The utility function could be deterministic or stochastic. For example, $u^i(s^i, s^{-i}) = 2$ with probability 0.7 and 0 otherwise. We denote the actual strategy selected by player i at round t by s_t^i and the strategy vector chosen by all other players by s_t^{-i}. We also denote the player i's received payoff in round t by u_t^i. To give an intuition, Table 1 shows a simple 2-player game. For example, for a round of the game in Table 1 (left) whose payoff is deterministic, if Player 1 plays strategy 1 and Player 2 plays strategy 2, they receive payoffs 8 and 2, respectively, in that round. For a round of the game in Table 1 (right) whose payoff is stochastic, if Player 1 plays strategy 1 and Player 2 plays strategy 2, they receive a fixed amount of payoffs (say 3) with probability 0.7 and 0.1, respectively, and receive a payoff of 0 otherwise.

A general concept used by many existing learning algorithms is attraction. Denoted by $A_t^i(j)$, the attraction reflects the propensity of player i towards the j^{th} strategy in round t. Intuitively, a larger attraction denotes a larger chance of the player selecting the strategy in question. An artificial agent probabilistically selects its strategy in each round according to the attractions of the strategies. Attractions can change in each round as players learn from the history and adjust. Table 2 summarizes some important notations in our paper.

To model human behaviors, we need to know the setting, i.e. what kind of information is available to each player. Such information includes the player's own strategy history, payoff history and payoff function, and the opponent's strategy, payoff history and payoff function. Given a game, different settings may lead to different human behaviors.

To evaluate the performance of a model, the most common metric is the behavior statistics of a game-playing process. As a repeated game is played for many rounds, we use the term "block" to denote a number of consecutive rounds. The whole game-playing process can be divided into blocks, where the behavior statistics are calculated for each block. The most typical behavior statistics include the following: 1) strategy rate: the observed proportion of times a player plays a certain strategy in each block. 2) efficiency: the observed expected payoffs in each block. 3) alternation: the observed proportion of times a player changes his strategy between two consecutive rounds.

To illustrate behavior statistics, Fig. 1 shows the proportion of choosing strategy 1 of each human player at each block of 40 rounds, regarding the game in Table 1 (left). This curve is called the *aggregate learning curve* which plots the behavior statistics for each block and is used by [24] for illustration of game-playing process and evaluation of artificial agents. We

Table 1 Example payoff functions of 2-player games. Left: deterministic payoff. Right: Stochastic payoff.

Player 1	Player 2	
	Strategy 1	Strategy 2
Strategy 1	3,7	8,2
Strategy 2	4,6	1,9
Strategy 1	0.3,0.5	0.7,0.1
Strategy 2	0.4,0.8	0.1,0.9

Please note that the numbers in the right table represent the probability of receiving a pre-specified payoff

Table 2 A list of important notations and their meanings

s^i, u^i	Variable representing strategy and payoff of player i, respectively
$S^{(i)}, U^{(i)}$	The set of possible strategies and payoffs of player i, respectively
$s_j^{(i)}, u_j^{(i)}$	The j^{th} strategy and its payoff of player i, respectively
s_t^i, u_t^i	The strategy and payoff, respectively, of player i at round t
s^{-i}	The combination of strategies of all players except player i
$s_{t,j}^i$	1 if $s_t^i = s_j^{(i)}$ and 0 otherwise
$A_t^i(j)$	Attraction of player i selecting the j^{th} strategy at round t
$P_t^i(j)$	Probability of player i selecting the j^{th} strategy at round t

also show the Nash equilibria for both players (0.875 and 0.375). We see that the behavior of human players does not converge to the Nash equilibrium. Our task is to predict these behavior statistics of each block. Evaluation is based on the deviation between the predicted curve and the ground-truth curve. To do this, we set up a simulator in which artificial agents play against each other. At each round, all agents simultaneously generate their strategies, and receive a corresponding payoff. Then, each agent updates its learning model based on the information it receives.

To sum up, building agents for simulating and predicting human behavior in game playing can be casted as a supervised learning problem.

The learning procedure is as follows:

1. Data collection. The dataset should include a specification of the game-theoretic environment as well as playing records of players. The game-theoretic environment describes the payoff function of each player and the observability of game-playing information.
2. Data preprocessing. We extract the behavior statistics at each block. Each game is played by many groups of human players. All behavior statistics in each block of each

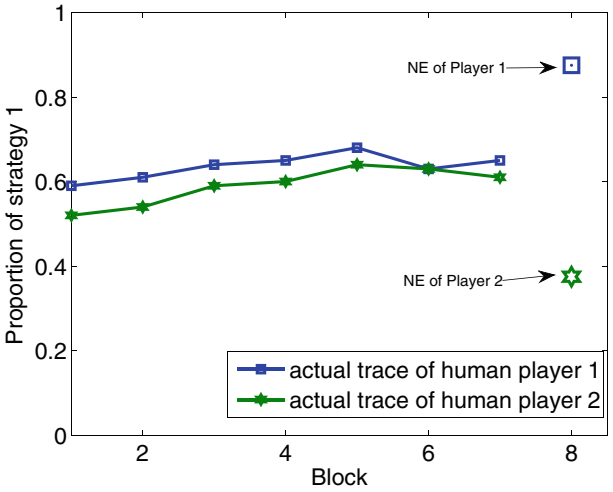

Fig. 1 Human playing traces and Nash equilibrium (NE) for the game in Table 1 *(left)*

game are calculated by averaging the ones of all groups playing it. Thus, the behavior statistics reflect human's aggregate behavior rather than individual ones.
3. Model building. Parameters in a learning model are selected to minimize the error between the predicted and ground-truth behavior statistics of each block. It is also possible to select other meta-parameters using cross-validation.

3 Related work

Two algorithms, fictitious play and reinforcement learning, form the basis of most existing agent learning methods in experimental economics. We review these two basic algorithms as well as other related variants and enhancements.

3.1 Fictitious play (FP)

In fictitious play [27], players are assumed to be able to observe their opponents' strategy, and play a best response to the past empirical frequencies of opponents' strategies. This is an example of belief learning in the sense that the frequencies can be viewed as beliefs by the player about his opponents, to which he responds. More formally, player i starts with an exogenous initial frequency function $\kappa_0^i : S^{-i} \rightarrow R^+$, and updates it as follows:

$$\kappa_{t+1}^i(s^{-i}) = \kappa_t^i(s^{-i}) + I(s_t^{-i}, s^{-i}), \tag{1}$$

where $I()$ is an indicator function, and $I(m, n) = 1$ if $m = n$ and 0 otherwise. Player i believes that the probability of player $-i$ playing strategy s^{-i} in round $t + 1$ is

$$\gamma_t^i(s^{-i}) = \frac{\kappa_t^i(s^{-i})}{\sum_{\hat{s}^{-i} \in S^{-i}} \kappa_t^i(\hat{s}^{-i})} \tag{2}$$

The player will then play the best response to this belief over opponents' strategy distribution next round, i.e.

$$P_{t+1}^i(j) = I(j, BR^i(\gamma_t^i(s^{-i}))) \tag{3}$$

where $P_{t+1}^i(j)$ is the probability player i plays strategy j at round $t + 1$, and the best response $BR^i(\gamma_t^i(s^{-i})) = \text{argmax}_k \sum_{s^{-i} \in S^{-i}} \gamma_t^i(s^{-i}) \cdot u^i(k, s^{-i})$.

The equations of fictitious play above could also be organized in a fashion of sequentially updating attractions and choosing the strategy with the greatest attraction at each round, as follows:

$$A_{t+1}^i(j) = A_t^i(j) + u^i(j, s_t^{-i}), \tag{4}$$

when $j = s_t^i$, $u^i(j, s_t^{-i})$ is equal to u_t^i which is the payoff player i received in round t. When $j \neq s_t^i$, $u^i(j, s_t^{-i})$ is called foregone payoff, which is the payoff when a player chooses other strategies while his opponents' strategies remain the same.[2] Thus, fictitious play can be applied only when the opponents' strategy history is visible or one's foregone payoff is available.

[2] With a slight abuse of notation, j and $s_j^{(i)}$ both refer to the j^{th} strategy.

3.2 Reinforcement Learning (RL)

The reinforcement learning [24] algorithm assumes that a player observes only his own payoff, and tracks behaviors rather than beliefs. Please note that the RL algorithms in this paper was proposed within the context of economics [24], rather than the traditional reinforcement learning in the context of AI [40]. Strategies are "reinforced" by their previous payoff. This is done by updating attractions. The chosen strategy is then a stochastic function of these attractions. Players who learn by reinforcement do not generally have beliefs about what other players will do. In basic RL, each artificial agent i starts from an initial attraction $A_0^i(j)$ for each strategy j, $j = 1, \cdots, m^i$, where m^i is the number of strategies available to player i. The initial attraction can be decided by schemes such as uniform initial attraction or cognitive hierarchy model. Then, the agent updates its attraction after each round of playing. After round t, upon receiving a payoff u_t^i, the attraction updating function is:

$$A_{t+1}^i(j) = A_t^i(j) + I(s_t^i, j)u_t^i \tag{5}$$

The probability of player i plays his j^{th} strategy is:

$$P_t^i(j) = \frac{A_t^i(j)}{\sum_{k=1}^{m^i} A_t^i(k)} \tag{6}$$

3.3 EWA

In general, psychologists focus more on reinforcement-based models such as RL while decision and game theorists emphasize on belief-based models such as FP. Thus, these two approaches have been regarded as fundamentally different [7]. A new algorithm called EWA (Experience-Weighted Attraction) combined the two and claimed itself as the first hybrid of both FP and RL. In the parametric EWA model [7], attractions are updated by

$$A_{t+1}^i(j) = \frac{\phi \cdot N(t) \cdot A_t^i(j) + [\delta + (1-\delta) \cdot I(j, s_t^i)] \cdot u^i(j, s_t^{-i})}{N(t) \cdot \phi \cdot (1-\kappa) + 1} \tag{7}$$

where $N(t)$ is updated by $N(t) = \phi \cdot N(t-1) \cdot (1-\kappa) + 1$, ϕ is a discount factor on previous experience, and δ is a discount factor on foregone payoff. Both ϕ and δ are between 0 and 1. κ is either 0 or 1, determining whether attractions would be normalized or not. The probability of player i using strategy j at round $t+1$ is updated by the following logistic response rule:

$$P_{t+1}^i(j) = \frac{e^{\lambda \cdot A_{t+1}^i(j)}}{\sum_{k=1}^{m^i} e^{\lambda \cdot A_{t+1}^i(k)}}, \tag{8}$$

where $\lambda \geq 0$ reflects the sensitivity to attraction. Since EWA requires to know the foregone payoff, it normally could only work in an environment where the foregone payoff is given in each round or both the self payoff matrix and opponents' strategy history are observable. A parametric EWA model has five free parameters to be learnt: $\lambda, \phi, \delta, A_0^i(j), \kappa$.

Self-tuning EWA [29], a state-of-the-art model, is an improvement to the EWA model. It allows all free parameters except for λ to be self tuned by heuristics. For example, self-tuning EWA uses the cognitive hierarchy model [8] to provide the initial attractions $A_0^i(j)$.

3.4 Other works

Game theory has been widely used by the multi-agent learning community. [39] proposed that two main goals of multi-agent learning models are being 1) computationally convenient for some properties of games such as Nash equilibrium, and 2) descriptive of the real human agents. Many attempts in the AI literature has been focusing on the first one: the efficient computation or analysis of Nash equilibrium strategy [1, 17, 31], or making agent as intelligent as possible [9, 12, 18]. However, few investigate the second aspect, i.e. are these models descriptive of real human behaviors? The key observation that human behaviors do not necessarily approach Nash equilibria is important in behavioral and experimental economics. There have been efforts to connect artificial intelligence with economics [5, 10, 11, 15], but few of them did research on learning human behaviors in repeated games. For example, [43] proposes a method for computing optimal defender strategies in a real-world security games against a quantal response (QR) attackers which reflects the bounded rationality of humans. However, they mainly investigate the computational issues of optimal strategies rather than learning human behaviors. There have been a few works of applying graphical model in game theory [30, 32, 34]. However, most of them deal with the issues of computing or approximating Nash Equilibrium solutions. Another line of research studies game theory in the context of online learning, which focuses more on the analysis of regret. The most prominent one is the multiplicative weights update method which applies a multiplicative updating rule to maintain weights over a set of experts [2, 33]. In this paper we mainly consider the additive updating rules which are the dominant choices in behavioral economics. Other related works include [14, 16] which models recursive reasoning of human. [36] also applies I-POMDP to model multi-agent system in repeated games. Our work is different in that we focus on repeated game and learning the aggregate learning curve of humans. [41] looks at two-player single-shot games where the actual utility value received by players is not observed. Instead, the utility features are observed whose linear combination is the utility received by players. They propose a method to recover the utility function from a set of samples from the joint strategy space. [42] also proposes a Bayesian framework for behavioral game theoretic models. But they focus on learning the parameters in existing models while our work involves explaining existing models as well as proposing new game-theoretic models within our framework.

In the literature of experimental economics, the dominating modeling methods are RL and FP. Based on RL and FP, variants include normalized reinforcement learning [22], averaging reinforcement learning [20], stochastic fictitious play [27], normalized fictitious play [26], self-tuning EWA [29], conditional fictitious play [28], moderated fictitious play [38], and other no-regret learning algorithms [27]. Since in this paper we focus on agent-based learning and decision-making models, some ensemble cognitive models such as I-SAW [19] will not be considered.

Table 3 shows the assumptions of various models. We can see that FP and EWA only work under some specific settings. RL can work under almost all kinds of settings, since each player always knows his submitted strategy and received payoff at each round. However, when some information such as the opponent's strategy at each round is available, RL-based algorithms fail to make use of that information. All these mentioned models only follow some rather simple rules, and are not suitable for modeling relatively intelligent human players. In this paper, we propose a novel unifying framework. First, our framework can handle a broad range of environmental settings. Second, this framework is able to derive existing models such as RL-like and FP-like models that follow simple rules, as well as

Table 3 Required information of various models. ⓪:player's payoff and strategy. ①: player's payoff function. ②:opponents' strategy. ③:opponents' payoff function. ④:player's foregone payoff

Model	Required information
Nash Equilibrium	① and ③
RL and its variants	⓪
FP and its variants	(⓪,① and ②) or (⓪ and ④)
EWA and its variants	(⓪,① and ②) or (⓪ and ④)

models exploring complicated game-playing patterns by inferring unavailable information from available information.

4 Sequential bayesian network: a unifying learning framework

Games in the real world are played under different settings. Existing learning approaches confine themselves to some specific settings. For example, as shown in Table 3, traditional FP and EWA agents can work only when the opponents' strategy history and the self-payoff function are known, or the player's forgone payoff is known. RL, on the other hand, does not require such information. However, even when such information (opponents' strategy and payoff) is available, RL cannot exploit such important data.

To study different artificial models of human behaviors, it is useful to have a framework broad enough to encompass both FP and RL, as well as important variations such as stochastic FP [27] and conditional FP [28]. In this paper, leveraging established graphical learning methods [3, 37], we propose a unifying framework to cover a wide range of settings in repeated games. This framework is based on a probabilistic graphical model with a sequential structure. For simplicity, we consider 2-player repeated games where there are various settings as follows (for the multi-player case, we can regard player 2 as the set of other players):

- The opponent's action is observable or unobservable.
- The opponent's payoff function is observable or unobservable.
- The self-payoff function is observable or unobservable.
- The payoff function is stochastic or deterministic.
- The payoff function is discrete or continuous.
- The strategy is discrete or continuous.

There are $2^6 = 64$ combinations. Though previous works might perform well in some specific settings, they discard useful information or even fail in some other cases. In contrast, leveraging the properties of graphical models, our framework can accommodate all these variations and make good use of available information.

Figure 2 shows our general graphical model, a sequential Bayesian network (SBN), for 2-player repeated games. The SBN is a standard Bayesian network with a sequential structure, and its nodes are assigned meanings as we will explain below. In order to prevent the abuse of notations and keep simplicity of presentation, we use player 1 and 2 rather than player i and $-i$ in 2-player games. Without loss of generality, we assume the learning

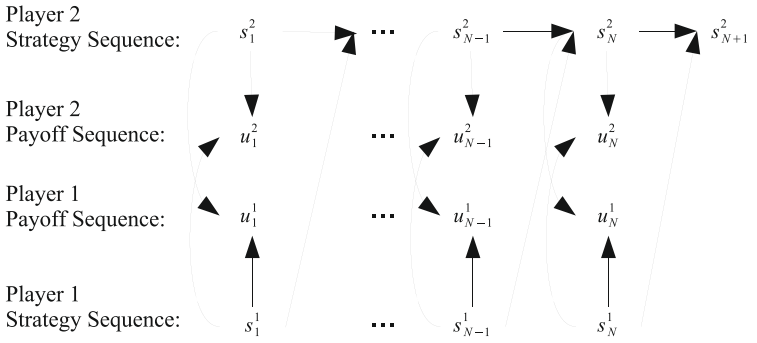

Fig. 2 A sequential Bayesian network (SBN) for learning in repeated 2-player games

model considers for player 1.[3] This model represents player 1's belief of his environment, including his reasoning process on the payoff function and the opponent's next possible strategy. Figure 2 follows the convention of graphical model where arrows in the model represent causal dependencies. Two dependencies here are important. First, player 1 believes that player 2's strategy at the next round depends on the strategy profile (combination of both players' strategies) at this round. Second, at each round the received payoff also depends on the strategy profile. These dependencies are modeled as arrows in the SBN model. In different game-theoretic settings, each player may not have full information of other players' strategy and payoff history, which means some nodes in Fig. 2 may be hidden. In this paper, all hidden nodes are colored white and all known nodes are colored gray. As a general framework, we color all nodes in Fig. 2 white. Below, we will show how nodes can be switched between observable and hidden in order to accommodate different game-playing settings.

Our framework addresses how to learn, update attraction and generate action for agents at each round. The procedure of our framework is to first predict the strategy distribution of player 2 in the next round, shown as the "s^2_{N+1}" in Fig. 2. Then, based on this inference, player 1 uses a "human response" to choose his own strategy in the next round. Possible types of human response are discussed later.

For the learning part, the SBN model tracks three beliefs for player 1 at each round (dependencies/arrows in Fig. 2):

- beliefs about player 2's strategy distribution at the next round, $P(s^2_{t+1}|s^1_t, s^2_t)$
- beliefs about player 1's utility function, $P(u^1|s^1, s^2)$
- beliefs about player 2's utility function, $P(u^2|s^1, s^2)$.

Traditional techniques in probabilistic graphical models can be used for tracking these three beliefs. If all random variables are observable, beliefs can be learned from the observable data by simply calculating sufficient statistics.

When some of them are unobservable, the EM algorithm [3, 13] can be used to infer it.

For attraction update, the SBN model first computes the expected utility (payoff) of each strategy based on the above beliefs. Denote the expected utility of player 1's j^{th} strategy at

[3]This means that each agent considers itself as player 1 and its opponent as player 2 while making inference.

round t by $\bar{u}_t^1(j)$. Let n_j denote the number of times that the j^{th} strategy has been used by player 1. We define attractions

$$A_t^1(j) = f(n_j)\bar{u}_t^1(j) \tag{9}$$

where $f(n_j)$ is a scaling factor and has output $f(n_j) = 1$ or n_j. When $f(n_j) = 1$, the attraction is exactly the expected payoff of each strategy in the history. When $f(n_j) = n_j$, the attraction of each strategy is equivalent to the accumulative payoff that strategy brings to the agent.

For action generation, the agent probabilistically selects one of its strategies according to their attractions. The probability of choosing a strategy at round t is given by one of the following possible "human response" schemes:

- Best response. Player 1 deterministically chooses the strategy with the highest attraction. This corresponds to a highly intelligent agent that maximizes expected utility.
- Average response. Player 1 chooses his j^{th} strategy with probability $\frac{A_t^1(j)}{\sum_k A_t^1(k)}$. Note that for this to make sense, attractions must be non-negative.
- Logistic response. Player 1 chooses his j^{th} strategy with probability

$$P_t^1(j) = \frac{e^{\lambda \cdot A_t^1(j)}}{\sum_k e^{\lambda \cdot A_t^1(k)}} \tag{10}$$

  The free parameter λ reflects the sensitivity to attraction. It provides a smoothing mechanism on the best response. For example, $\lambda = \infty$ gives the best response, corresponding to intelligent players; while $\lambda = 0$ gives a random response, corresponding to players that are totally indifferent to all strategies.

In summary, the SBN learning framework consists of three steps at each round of a repeated game:

1. Learn beliefs. Use statistical inference such as maximum-a-posteriori (MAP) estimation, maximum likelihood (ML) estimation, or posterior inference (PI) to update all beliefs in the model to accommodate player 1's new observations.
2. Update attraction. Calculate the expected payoff of each strategy based on the beliefs. The attraction of each strategy equals to the expected payoff $\bar{u}_j^1$ times a scaling factor $f(n_j)$.
3. Generate actions. Choose a response scheme to determine player 1's strategy in the next round.

Note that SBN is a general model which can be reduced into simpler models. Special cases for different settings can be derived from SBN by removing dependencies or marking variables as hidden or observable. All models under the SBN framework can follow the same inference process. As a result, the SBN framework provides a comprehensive foundation for learning human behaviors in repeated games.

For multi-player games, we can still utilize the SBN framework by changing the player 1 to player i and player 2 to player $-i$ in Fig. 2. In multi-player games, player $-i$ is a combination of all players except i rather than just one player, and so do s^{-i} and u^{-i}. In this way, the opponents of player i are aggregated and act as an "environment" of player i. Thus, multi-player games are reduced to 2-player games where each player is faced with his environment.

5 Bayesian explanation of previous models

We show that our SBN framework provides a unifying Bayesian explanation to the two major approaches of agent learning: reinforcement learning and fictitious play.

5.1 Reinforcement learning

Reinforcement learning can be viewed as a special case of SBN by removing the variables for the opponent's payoff and strategy sequences as well as their incident edges, which means that the decision of player 1 is independent of the opponent's strategy. The graphical representation of reinforcement learning is shown in Fig. 3. We color hidden nodes in white and observable nodes in light gray. Reinforcement learners do not learn the strategy distribution of opponents, but directly infer the expected utility.

Theorem 1 *Reinforcement learning is equivalent to the sequential Bayesian network in* Fig. 3 *with maximum likelihood inference.*

Proof Given player 1's strategy and payoff history in the first N rounds, maximum likelihood (ML) inference on the Bayesian network in Fig. 3 yields an optimization problem:

$$\max \prod_{t=1}^{N} P(u_t^1|s_t^1) = \prod_{t=1}^{N} \prod_{i=1}^{|U^{(1)}|} \prod_{j=1}^{|S^{(1)}|} P(u_i^{(1)}|s_j^{(1)})^{u_{t,i}^1 s_{t,j}^1} \tag{11}$$

$$\text{subject to: } \sum_{i=1}^{|U^{(1)}|} P(u_i^{(1)}|s_j^{(1)}) = 1, \text{ for each } s_j^{(1)} \in S^{(1)}$$

where $u_{t,i}^1 = I(u_t^1, u_i^{(1)})$ and $s_{t,j}^1 = I(s_t^1, s_j^{(1)})$. Obviously, changing the objective function by taking logarithm of (11) will not change the solution. Thus, using Lagrange multipliers on the logarithm of (11), the objective function becomes:

$$E = \sum_{t=1}^{N} \sum_{i=1}^{|U^{(1)}|} \sum_{j=1}^{|S^{(1)}|} u_{t,i}^1 s_{t,j}^1 \log P(u_i^{(1)}|s_j^{(1)}) - \mu\Big(\sum_{i=1}^{|U^{(1)}|} P(u_i^{(1)}|s_j^{(1)}) - 1\Big) \tag{12}$$

Take the derivative of E over $P(u_i^{(1)}|s_j^{(1)})$ and make it equal to 0, we get:

$$P(u_i^{(1)}|s_j^{(1)}) = \frac{\sum_{t=1}^{N} u_{t,i}^1 s_{t,j}^1}{\mu} \tag{13}$$

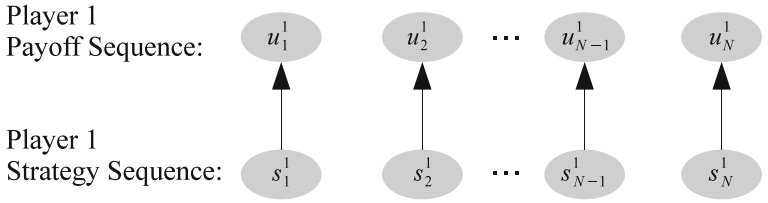

Fig. 3 A graphical representation of reinforcement learning derived from SBN

Combining with $\sum_{i=1}^{|U^{(1)}|} P(u_i^{(1)}|s_j^{(1)}) = 1$, we have:

$$\mu = \sum_{i=1}^{|U^{(1)}|}\sum_{t=1}^{N} u_{t,i}^1 s_{t,j}^1 = \sum_{t=1}^{N}\sum_{i=1}^{|U^{(1)}|} u_{t,i}^1 s_{t,j}^1 = \sum_{t=1}^{N} s_{t,j}^1 \sum_{i=1}^{|U^{(1)}|} u_{t,i}^1 = \sum_{t=1}^{N} s_{t,j}^1 \tag{14}$$

Recall that n_j is the number of times strategy j has been used by player 1. Thus, we have $\sum_{t=1}^{N} s_{t,j}^1 = n_j$, since $s_{t,j}^1 = I(s_t^1, s_j^{(1)})$. Combining with (13), we have:

$$P(u_i^{(1)}|s_j^{(1)}) = \frac{\sum_{t=1}^{N} s_{t,j}^1 u_{t,i}^1}{n_j} \tag{15}$$

Thus, we can calculate the attraction for strategy $s_j^{(1)}$ at round $N+1$ according to (9), as follows:

$$\begin{aligned} A_{N+1}^1(j) &= f(n_j)\sum_{i=1}^{|U^{(1)}|} u_i^{(1)} P(u_i^{(1)}|s_j^{(1)}) \\ &= f(n_j)\frac{\sum_{t=1}^{N} s_{t,j}^1 \sum_{i=1}^{|U^{(1)}|} u_{t,i}^1 u_i^{(1)}}{n_j} \\ &= f(n_j)\frac{\sum_{t=1}^{N} s_{t,j}^1 u_t^1}{n_j} \end{aligned} \tag{16}$$

where $\sum_{i=1}^{|U^{(1)}|} u_{t,i}^1 u_i^{(1)} = \sum_{i=1}^{|U^{(1)}|} I(u_t^1, u_i^{(1)}) u_i^{(1)} = u_t^1$. (16) is exactly the rule for reinforcement learning that reinforces selected strategy by its received payoff. □

If $f(n_j) = 1$ and the player adopts the average response, then it is averaging reinforcement learning (ARL) [20]. If $f(n_j) = n_j$, then (16) can be transferred to

$$A_{N+1}^1(j) = \sum_{t=1}^{N} s_{t,j}^1 u_t^1 = A_N^1(j) + s_{t,j}^1 u_t^1 = A_N^1(j) + I(s_t^1, s_j^{(1)}) u_t^1 \tag{17}$$

which is exactly the same as the standard RL in (5). When the player adopts the average response shown in (6), then this type of SBN becomes basic accumulative reinforcement learning (BRL) [24]. Hence, we provide an insight that reinforcement learning can actually be interpreted by a graphical Bayesian model.

5.2 Fictitious play

Another important special case of SBN is when player 1's payoff function and player 2's strategy history is not hidden. This case can also be modeled by SBN as shown in Fig. 4, in which when a player can observe the opponent's strategy as a Markov chain. Note that player 1's strategy and payoff sequences are removed from graph, since his payoff function is known and thus will not be included in the inference process.

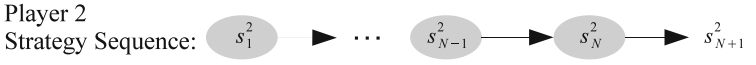

Fig. 4 A graphical representation of fictitious play derived from SBN

Theorem 2 *Fictitious play is equivalent to the sequential Bayesian network in* Fig. 4 *when best response is used.*

Proof Since all the parameters are known in this case, we can strengthen this model by allowing high-order Markov dependencies and prove a more general result. For a k-th order Markov chain ($k = 0, 1, ...$), using maximum-a-posteriori (MAP) inference on the model in Fig. 4, we have the following formula for player 2's strategy transition probability by sufficient statistics [3]:

$$P(s|s^{\leftarrow k}) = \frac{n(s^{\leftarrow k}s) + \alpha(s^{\leftarrow k}s)}{\sum_{s\in S^2} n(s^{\leftarrow k}s) + \sum_{s\in S^2} \alpha(s^{\leftarrow k}s)} \tag{18}$$

where $n(s)$ is the number of times a sequence s appears in player 2's strategy history, $s^{\leftarrow k}$ indicates a length-k sequence of strategies and α is the conjugate Dirichlet prior. This is exactly the k-th order conditional fictitious play (CFP) [28], a generalization of FP. When $k = 0$ (which means that the strategy at this round does not depend on the strategies in the history and (18) would be the same as (2)) and a player adopts the best response (as shown in (3)), we get exactly the standard FP [27]. □

More special cases can be derived from this model:

- When $k = 0$ but a player adopts the logistic response, we get the stochastic fictitious play (SFP) [27].
- When we use the best response and the first-order Markov property ($k = 1$) but assume the transition probability from a strategy s to any other strategy s' to be 0, i.e. $P(s|s) = 1$ and $P(s'|s) = 0$, $s' \neq s$, we get the Cournot adjustment process (CAP) [27].
- Following the same setting as SFP, if posterior inference (PI) instead of MAP inference is used, we get the moderated fictitious play (MFP) [38].

To summarize, we compare all the mentioned models in Table 4.

Table 4 Comparison of artificial agents that can be derived from the general SBN framework.

model	response type	opponent's strategy distribution	visibility of opponent's strategy	$f(n_j)$	inference
BRL	average response	independent	not required	n_j	ML
ARL	average response	independent	not required	1	ML
FP	best response	multinomial	required	1	MAP
SFP	logistic response	multinomial	required	1	MAP
MFP	logistic response	multinomial	required	1	PI
CFP	best response	Markovian	required	1	MAP
CAP	best response	repeating last action	required	1	MAP
HMP	logistic response	Markovian	not required	1	ML

6 Hidden markovian play: an agent with limited observable information

The SBN framework not only provides a unifying explanation to existing RL and FP agents, but also enables us to derive new agents. In this section, we derive a learning model from the SBN framework to handle an important game-playing setting which cannot be handled by FP and EWA agents.

In the real world, agents interact with each other with limited observable information. In many cases, an agent submits his strategy and receives a payoff in each round, without knowing the opponent's strategy history or payoff function. He may even be unaware of his own payoff function. Traditional FP and EWA models do not work under this setting. The RL model survives in this setting, but it does not make the most out of all available information. We want to develop models for the setting in which:

1. Each player does not know his opponent's strategy history, payoff history or payoff function; and
2. Each player does not know his own payoff function. He only knows his submitted strategy and received payoff in each round.

To cope with this setting, we use a special case of our general SBN model called the Hidden Markovian Play (HMP). HMP learns to construct the payoff function of the player himself and infer the opponent's strategy history. Figure 5 shows the graphical representation of HMP which can be viewed as a graphical combination of RL (See Fig. 3) and FP (See Fig. 4). The belief model of player 1 part comes from RL and the player 2 part comes from FP except for that the strategy sequence of player 2 is unobservable. Thus, HMP is a hybrid model incorporating both RL and FP from the perspective of the Bayesian network. The intuition is that, while agents try to maximize their payoff in the next round based on their own history as in RL, HMP allows agents to consider what potential strategy the opponent would likely choose for the next round. Human players are not absolute RL thinkers or FP thinkers. By doing this combination, the HMP agent represents a more sophisticated model with stronger learning ability. In this model, player 1 maintains two beliefs: the belief about his self-payoff function and the belief about player 2's strategy transition probability.

Formally, the parameters we want to infer are $\theta = (\pi, B, T)$ where

- π is player 2's initial distribution over his strategy space.
- T is player 1's belief about the strategy transition probability of player 2. T_{ij} is the transition probability of player 2 from choosing strategy $s_i^{(2)}$ to $s_j^{(2)}$, i.e. $T_{ij} = P(s_j^{(2)}|s_i^{(2)})$

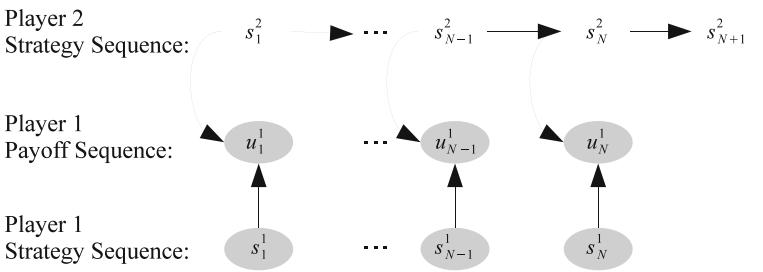

Fig. 5 A graphical representation of hidden Markovian play derived from SBN

– B is player 1's belief about his own payoff function:

$$B = \{b_{ij}(u_k^{(1)})\} = \{P(u_k^{(1)}|s_i^{(1)}, s_j^{(2)})\}.$$

We use maximum likelihood for parameter inference. Since this model contains hidden variables, as specified in our inference procedure, we use the EM algorithm [3, 13] to maximize the likelihood of the received payoff sequence given the submitted strategy sequence.

We now discuss the case where all parameters are discrete. The case with continuous parameters can be handled similarly. In the discrete case, we assume that they take the form of multinomial distributions, and use the 1-of-K coding scheme: for a multinomial variable z_i, if $z_{ik} = 1$, then z_i takes its k^{th} state. Note that $\sum_{k=1}^{K} z_{ik} = 1$, which means that z_i takes one and only one state. Denote player i's strategy sequence by $S^i = \{s_t^i | 1 \le t \le N\}$ and player 1's payoff sequence by $U^1 = \{u_t^1 | 1 \le t \le N\}$.

Denote the marginal posterior distribution of the hidden strategy s_t^2 by $\gamma(s_t^2) = P(s_t^2|S^1, U^1, \theta)$ and the marginal posterior distribution of two successive latent strategies by $\xi(s_{t-1}^2, s_t^2) = P(s_{t-1}^2, s_t^2|S^1, U^1, \theta)$. In the E step, we have

$$\gamma(s_t^2) = \frac{\alpha(s_t^2) \cdot \beta(s_t^2)}{P(U^1|S^1, \theta)}, \tag{19}$$

$$\xi(s_{t-1}^2, s_t^2) = \frac{\alpha(s_{t-1}^2) \cdot b_{s_t^1, s_t^2}(u_t^1) \cdot T_{s_{t-1}^2, s_t^2} \cdot \beta(s_t^2)}{P(U^1|S^1, \theta)}, \tag{20}$$

where

$$\alpha(s_t^2) = P(u_1^1, ..., u_t^1, s_t^2|S^1), \tag{21}$$

$$\beta(s_t^2) = P(u_{t+1}^1, ..., u_N^1|s_t^2, S^1), \tag{22}$$

where $\alpha(s_t^2)$ and $\beta(s_t^2)$ can be figured out by a forward-backward algorithm [3] that is similar to the one for HMM:

$$\alpha(s_t^2) = b_{s_t^1, s_t^2}(u_t^1) \sum_{s_{t-1}^2 \in S^{(2)}} \alpha(s_{t-1}^2) \cdot T_{s_{t-1}^2, s_t^2}, \tag{23}$$

$$\beta(s_t^2) = \sum_{s_{t+1}^2 \in S^{(2)}} \beta(s_{t+1}^2) \cdot b_{s_{t+1}^1, s_{t+1}^2}(u_{t+1}^1) \cdot T_{s_t^2, s_{t+1}^2}. \tag{24}$$

In the M step, we have:

$$\pi_k = \frac{\gamma(s_{1k}^2)}{\sum_{j=1} \gamma(s_{1j}^2)}, \tag{25}$$

$$T_{ij} = \frac{\sum_{t=2}^{N} \xi(s_{t-1,i}^2, s_{t,j}^2)}{\sum_k \sum_{t=2}^{N} \xi(s_{t-1,i}^2, s_{t,k}^2)}, \tag{26}$$

$$b_{ij}(u_k^{(1)}) = \frac{\sum_{t=1}^{N} \gamma(s_{tj}^2) s_{ti}^1 u_{tk}^1}{\sum_{t=1}^{N} \gamma(s_{tj}^2) s_{ti}^1}. \tag{27}$$

Intuitively, in each round, player 1 first makes inference on his own payoff function, the opponent's strategy history, and then the strategy transition probability, based on his own sequences of submitted strategies and received payoffs. The inference process repeats the EM iterations until convergence. Then, player 1 makes inference on the opponent's strategy distribution for the next round based on the opponent's strategy transition probability:

$$P_{N+1}^2(j) = \sum_{s_N^2 \in S^{(2)}} \gamma(s_N^2) T_{s_N^2, j} \tag{28}$$

The next step is to calculate player 1's strategy attraction based on the constructed payoff function:

$$A^1_{N+1}(j) = \sum_k \{P^2_{N+1}(k)[\sum_{u^{(1)}} u^{(1)} \cdot b_{jk}(u^{(1)})]\} \tag{29}$$

Here, the attraction of each strategy is the expected payoff by setting $f(n_j) = 1$ in (9). Finally, player 1 chooses his strategy for the next round by logistic response in (10) based on these attractions. The process is efficient and takes polynomial time, since it exploits the sequential structure of the SBN framework.

In terms of multi-player games, player 2 is replaced by player $-i$ which is a combination of all opponents of player i, as shown in Section 4. s^{-i} does not have to be the product of strategy spaces of all player i's opponents, since s^{-i} can be aggregated in most cases. For example, in n-player market entry games, payoff of player i is decided by the total number of players in the market rather than a specific strategy selected by each opponent. Thus, s^{-i} can be represented by the number of players in the market except player i, which only requires n states in total.

From Table 4, we can realize that there are still lots of models yet to be explored. For example, we could reduce from HMP to a "hidden fictitious play" by assuming a multinomial distribution for the opponent's strategy. We could also investigate the accumulative effect of payoff by assigning $f(n_j) = n_j$. Therefore, SBN is versatile in providing solutions for various settings with a unified inference procedure.

7 Experimental evaluation

In this section, we evaluate the performance of HMP, and compare it with several popular existing artificial game-playing agents in the literature. We apply these agents to three real-world problems, including a one-player gamble game, a two-player strategic game, and a multi-player Market Entry Game modeling the real market. All the data for these three games are collected from real-world experiments by human participants.

7.1 Experimental setup

For comparison, we implement the following popular existing agents: a naive agent that randomly makes a decision with an equal probability for each strategy (Random), basic reinforcement learning (RL)[24], normalized reinforcement learning (NRL)[22], fictitious play (FP)[27], stochastic fictitious play (SFP)[27], normalized fictitious play (NFP)[26], parametric Experience Weighted Attraction (EWA)[7], and self-tuning Experience Weighted Attraction (STEWA)[29]. For each agent, to test its performance, we set all players in a game-playing session to be using the same agent.

The proposed HMP model works under many settings as it requires minimum information comparing to others. The first and second datasets are in a game-theoretic environment with minimum feedback, where only a player's own strategy and payoff are observable. Thus, only Random, RL, NRL and HMP can be applied to these two games. In the third game (Market Entry Game), foregone payoff is also observable and hence all the above agents can be applied.

For the implementation of the HMP model, we set a memory length of 6 for each HMP player. That is, as the game moves forward, players only remember their payoff and strategy of the previous 6 rounds. One reason for this implementation is to simulate the real-world

scenario as human players have limited memory length and they cannot memorize the outcomes of all rounds. From the perspective of computation, a short memory length also dramatically speeds up the inference procedure of the HMP model.

The purpose of this evaluation aims to predict the aggregate human behavior instead of individual ones. Given a game, each agent model is run 200 times. The predicted behavior statistics in each block is the average of the ones in all 200 runs in the corresponding block.

Each agent above has its own free parameters whose values are yet to be decided. Agents are evaluated by MSD (Mean Square Deviation) of a m-fold cross validation between predicted and observed behavior statistics. That is, we divide a dataset into m subsets with 1 subset for testing and the remaining $m-1$ subsets for training. All free parameters in a model are tuned to best fit the training set and then tested on the validation set. The selection of parameter values is based on a grid search. This process is repeated m times with each subset used exactly once as the validation set. The MSD of the m-fold cross validation is $\sum_{i=1}^{m} D_i^2/m$, where D_i is the deviation when the i^{th} subset is used as the validation set.

The block size and behavior statistics of each type of games are already decided in the dataset. For games with single type of behavior statistics, we use the mean, standard deviation and maximum of MSD in cross-validation. For games with multiple type of behavior statistics, we use NMSD (Normalized MSD) instead, which normalizes the MSD for each type of behavior statistics. The normalization factor is also provided by the dataset.

7.2 One-player gamble games

The first dataset is from the Technion Prediction Tournament [22], which provides real data on how humans make decision from experiences. One hundred participants joined the data collection process. They were instructed to individually play a repeated gamble game (100 rounds per game) where there are two available options for the player in each round – safe or risky gamble. If the safe option is taken, the player could win M dollars for sure. If the risky option is taken, the player could win H dollars with probability Ph or L otherwise ($H \geq M \geq L$).

$$u_t^1 = \begin{cases} M & \text{safe option} \\ H \text{ with probability } Ph \text{ and } L \text{ otherwise} & \text{risky option} \end{cases}$$

The dataset contains 40 games, each of which has different values of (H, M, L, Ph). The block size of each game is 100 rounds, which means that the behavior statistics are calculated in terms of the whole game. We use the term "risk rate" to denote the percentage of the "risky" choices a player takes for each individual game. Performance of each agent is evaluated by the MSD between the model-predicted risk rate with the true risk rate in the data. Evaluation is based on a 10-fold cross validation. Table 5 shows the evaluation results. We can see that the HMP agent outperforms all the other agents. It not only gives the lowest mean across all the games, but also has the lowest standard deviation and maximum MSD.

Table 5 The mean, standard deviation, and maximum of MSDs ($\times 10^2$) on one-player gamble games

	Random	BRL	NRL	HMP
mean	4.28	4.48	1.95	1.38
std. dev.	1.40	1.73	0.65	0.55
max	7.22	8.57	3.04	2.25

7.3 Two-player games

In the literature of virtual economics, the most common case is 2×2 matrix normal-form games, where each player has two strategies and each element in the matrix corresponds to a payoff. To make things more realistic and general, we test our agents on a real dataset whose matrix element corresponds to the probability of winning a certain payoff at each round. The dataset is published in [25]. This dataset contains 10 games, each of which consists of 500 rounds. The block size is 100 rounds for this dataset. We use $pp^i(j,k)$ to denote the predicted probability of player i selecting the first choice at block j in game k and $gp^i(j,k)$ to denote the ground-truth probability. Performance of each agent is evaluated by the MSD of a 10-fold cross validation. Specifically, the MSD for a game set GS is calculated as follows:

$$\text{MSD} = \frac{\sum_{g \in GS} \sum_{i=1}^{2} \sum_{j=1}^{5} (pp^i(j,g) - gp^i(j,g))^2}{10 * |GS|}$$

Results on Table 6 show that HMP is significantly better than all the other agents. Since each game in this dataset contains 5 blocks which is long enough to form a curve, we randomly pick one game and plot its aggregate learning curve [24] for visualization, as shown in Fig. 6. This aggregate learning curve shows the behavior statistics (rate for selecting choice 1) in each block, which clearly presents the playing process of human players (ground truth), NRL agent, and HMP agent We can see that HMP mimics the human behavior more closely than NRL does.

7.4 Multi-player market entry games

We also test the agents on Market Entry Games whose data is from the Market Entry Prediction Competition [21]. This kind of games have important applications in economics as they give a description of how a market operates especially the effect that too many vendors entering into a market would cause oversupply. There are in total 40 market entry games of 50 rounds per game in this dataset, provided by the CLER Lab at Harvard University. Games are repeated 4-person market entry games with environmental uncertainty. At each round of these games, each player has to decide whether to enter a risky market or stay out. The payoff player i receives at round t is

$$u_t^i = \begin{cases} 10 - r * E + G_t & \text{if player } i \text{ enters} \\ \pm G_t / W,\ P(+) = P(-) = 0.5 & \text{if player } i \text{ stays out} \end{cases}$$

where G_t is a realization of a binary gamble at round t. $G_t = H$ with probability Ph and $G_t = L$ with probability $1 - Ph$, and $Ph = \frac{-L}{H-L}$ ($H > 0$ and $L < 0$). E is the number of entrants. Two other parameters r and W are integers greater than 1. Hence, each specific game is characterized by the parameters $\theta = (r, W, H, L)$.

In these games, the foregone payoff is available, and all agents including EWA and FP can be applied.

Table 6 The mean, standard deviation, and maximum of MSDs ($\times 10^2$) on two-player games

	Random	BRL	NRL	HMP
mean	3.42	1.74	0.86	0.61
std. dev.	2.14	1.37	0.92	0.81
max	6.87	3.67	3.33	2.88

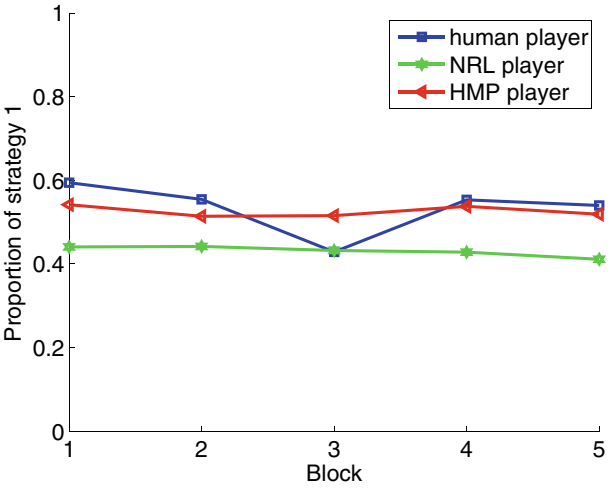

Fig. 6 Aggregate learning curves on a 2-player game

For evaluation, we use a 10-fold cross validation on these 40 market entry games. The evaluation of agents are based on NMSD (Normalized Mean Squared Deviation) which consists of three behavior statistics in each block of 25 rounds: entry rate, efficiency and alternation. Thus, there are in total 6 statistics for each game as it is played for 50 rounds. The entry rate at a block is the percentage of participants entering into the market. The efficiency is the observed expected payoff of all players at a block. The alternation is the proportion of times that players alter their strategies between rounds at one block. We use $per(i, g)$, $pe(i, g)$ and $pa(i, g)$ to denote these predicted entry rate, efficiency, and alternation at block i in game g, respectively; and use $ger(i, g)$, $ge(i, g)$ and $ga(i, g)$ to denote the ground-truth statistics. To compute NMSD, each statistics is weighted by their estimated error variances provided by this dataset. We use $eer(i)$, $ee(i)$ and $ea(i)$ to denote the estimated error variance of these behavior statistics. The NMSD is the average of these 6 weighted statistics:

$$\text{NMSD} = \frac{\sum_{g \in GS} \sum_{i=1}^{2} [\frac{(per(i,g)-ger(i,g))^2}{eer(i)} + \frac{(pe(i,g)-ge(i,g))^2}{ee(i)} + \frac{(pa(i,g)-ga(i,g))^2}{ea(i)}]}{6 * |GS|}$$

Table 7 shows the evaluation results. We observe that HMP performs the best among all 9 agent models, with EWA being the second best. All other agents are significantly worse than HMP and EWA. HMP has not only lower mean NMSD, but also lower variance and maximum, indicating a consistent and robust performance improvement. Although EWA also performs well, the number of free parameters of EWA is 5 which is the largest among all models, as shown in Table 7. Having so many free parameters largely slows down the training process and has a potential risk of losing generalization ability on other datasets. In contrast, HMP has only one free parameter (λ).

Table 7 The mean, standard deviation, and maximum of NMSDs on the Market Entry Games. (N_f is the number of free parameters in each model)

	Random	BRL	NRL	FP	SFP	NFP	EWA	STEWA	HMP
mean	32.39	13.34	10.39	11.78	11.33	8.099	5.296	10.80	5.275
std. dev.	10.87	7.850	4.425	3.277	2.748	2.597	1.868	7.131	1.586
max	48.50	32.25	18.73	17.42	16.57	10.24	8.636	27.82	8.263
N_f	0	1	2	1	2	2	5	1	1

7.5 Sensitivity to λ

The λ in logistic response as shown in (10) is the only free parameter that requires to be tuned in our HMP agent. Figure 7 shows the sensitivity of the HMP agent to the value of λ in the above three datasets. We could see that the MSD and NMSD goes down and then up as the value of λ increases, leading to an optimal value in the middle. When λ is very small, the model is relatively sensitive to its value. In fact, when $\lambda = 0$, HMP is exactly the same as the random response model.

When λ is reasonably large, HMP is not sensitive to λ. The reason is that the probability of choosing one action at each round would be dominated by the largest action, which becomes a best response, since λ is an exponential component when calculating the probability of choosing strategies. To illustrate the point further, in Fig. 7, for 1-player and 2-player games, we plot a dotted line to show the MSD of NRL; for market entry games, we plot a dotted line to show the NMSD of EWA. We can choose any value of λ in the region that is below the dotted line and still obtain better modeling performance than NRL or EWA. As we can see from Fig. 7, the range of such λ is quite large, showing that HMP can perform well for a large range of λ, avoiding the need for extensive tuning. We can also observe from Fig. 7 that if λ is fixed at 4.0 across all the 130 games, the performance of HMP is only a little bit worse than choosing the optimal λ for each category of games. Thus, due to the insensitivity of the performance of the HMP agent to λ, we can use a rule of thumb to choose a reasonably good value of λ ($\lambda = 4.0$ in our case). When being used in this way, the HMP model does not have any free parameter and does not require any tuning. Such a parameter-free model can be a good choice for situations when the training time is limited.

Another interesting point is that the value of λ that best fits a dataset, to some extent, reflects the expected intelligence of people in that dataset, as the larger λ is, the more intelligent the HMP agent will be.

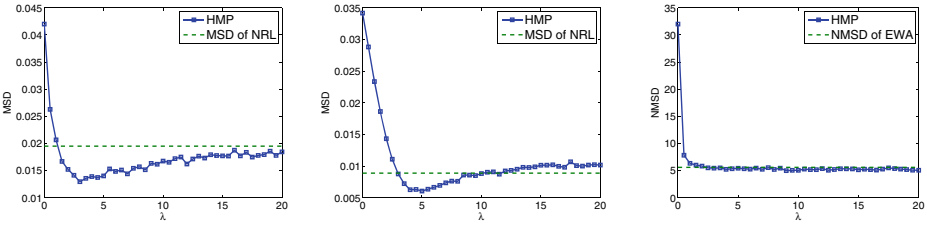

Fig. 7 HMP's sensitivity to λ. From left to right:1-player gamble game, 2-player game and Market Entry Game

8 Conclusions

We have presented a general machine learning framework for building artificial agents to simulate human behaviors in playing games. Although computational economists have developed various game-playing agents, they have not used well-established machine learning methods such as graphical models. We have proposed a sequential Bayesian network (SBN) framework to bridge this gap. The SBN framework provides deep insights which unify two fundamental agent learning approaches, reinforcement learning and fictitious play, in a general Bayesian inference process. The graphical representation gives a clear visualization on the playing and reasoning process in repeated games. Moreover, from the SBN framework, we have derived Hidden Markovian Play (HMP), a new agent that does not require observability of a player's own payoff function, or the opponent's strategy and payoff.

Experimental results show that the HMP agent consistently outperforms many other existing agents on real data of humans playing a variety of games. Moreover, with only one free parameter, the HMP agent actually obtained better performance using fewer free parameters than all other agents, making the results even more impressive. Experimental results have also shown that the performance of the HMP agent is robust under changes of the only free parameter λ. Our work bridges the gap between computational economics and machine learning, and makes a substantial step towards building virtual economics, which potentially opens up new applications of machine learning in economics.

Acknowledgments This work is supported by CCF-1215302 and IIS-1343896 from the National Science Foundation of the US.

References

1. Airiau, S., Endriss, U.: Multiagent resource allocation with sharable items: Simple protocols and nash equilibria In: Proceedings of the 9th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2010) (2010)
2. Arora, S., Hazan, E., Kale, S.: The multiplicative weights update method: a meta-algorithm and applications. Theory Comput. **8**(1), 121–164 (2012)
3. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer-Verlag, New York, Inc. Secaucus, NJ (2006)
4. Bonabeau, E.: Agent-Based Modeling: Methods and Techniques for Simulating Human Systems. Proc. Natl. Acad. Sci. U. S. A. **99**(10), 7280–7287 (2002)
5. Boutilier, C., Shoham, Y., Wellman, M.P.: Economic principles of multi-agent systems. Artif. Intell. **94**(1–2), 1–6 (1997)
6. Bowling, M., Veloso, M.: Existence of multiagent equilibria with limited agents. J. Artif. Intell. Res. **22**(1), 353–384 (2004). http://dl.acm.org/citation.cfm?id=1622487.1622498
7. Camerer, C., hua Ho, T.: Experience-weighted attraction learning in normal form games. Econometrica **67**, 827–874 (1999)
8. Camerer, C.F., Ho, T.H., Chong, J.K.: A cognitive hierarchy model of games. Q. J. Econ. **119**(3), 861–898 (2004)
9. Carmel, D., Markovitch, S.: Learning models of intelligent agents. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence pp 62–67 Portland Oregon (1996)
10. Chen, Y., Lai, J., Parkes, D.C., Procaccia, A.D.: Truth, justice, and cake cutting. In: Fox, M., Poole, D. (eds.): AAAI 2010, Atlanta, Georgia, USA, July 11–15, 2010. AAAI Press (2010)
11. Chen, Y., Vaughan, J.W.: A new understanding of prediction markets via no-regret learning: In: Proceedings of he 11th ACM conference on Electronic commerce, EC '10 pp 189–198. ACM, New York, NY (2010)
12. Crandall, J.W., Ahmed, A., Goodrich, M.A. In: Burgard, W., Roth, D. (eds.): Learning in repeated games with minimal information: The effects of learning bias. AAAI. AAAI Press (2011)

13. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum Likelihood from Incomplete Data via the EM Algorithm. J. Royal Statist. Soc. Series B (Methodological) **39**(1), 1–38 (1977)
14. Devaine, M., Hollard, G., Daunizeau, J.: Theory of mind: did evolution fool us. PloS. One. **9**(2), e87619 (2014)
15. Dimicco, J.M., Greenwald, A., Maes, P.: Learning curve: A simulation-based approach to dynamic pricing. J. Electron. Commer. Res. **3**, 245–276 (2003)
16. Doshi, P., Qu, X., Goodie, A., Young, D.: Modeling recursive reasoning by humans using empirically informed interactive pomdps In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1 - Volume 1, AAMAS '10, pp.1223–1230. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2010)
17. Elkind, E., Golberg, L.A., Goldberg, P.W.: Computing good nash equilibria in graphical games Proceedings of the 8th ACM conference on Electronic commerce, EC '07, pp. 162–171. ACM, New York, NY USA (2007)
18. Elkind, E., Leyton-Brown, K.: Algorithmic game theory and artificial intelligence. Artif. Intell. Mag. **31**(4), 9–12 (2010)
19. Erev, I.: On surprise, change, and the effect of recent outcomes. Front. Psychol. **3**(0) (2012)
20. Erev, I., Bereby-Meyer, Y., Roth, A.E.: The effect of adding a constant to all payoffs: experimental investigation, and implications for reinforcement learning models. J. Econ. Behav. & Organ. **39**(1), 111–128 (1999)
21. Erev, I., Ert, E., Roth, A.E.: A choice prediction competition for market entry games: An introduction. Games **1**(2), 117–136 (2010). doi:10.3390/g1020117
22. Erev, I., Ert, E., Roth, A.E., Haruvy, E., Herzog, S., Hau, R., Hertwig, R., Steward, T., West, R., Lebiere, C.: A choice prediction competition, for choices from experience and from description. J. Behav. Decis. Mak. **23**, 15–47 (2010)
23. Erev, I., Roth, A., Slonim, R., Barron, G.: Learning and equilibrium as useful approximations: Accuracy of prediction on randomly selected constant sum games. Econ. Theory **33**(1), 29–51 (2007)
24. Erev, I., Roth, A.E.: Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria. Am. Econ. Rev. **88**(4), 848–81 (1998)
25. Erev, I., Roth, A.E., Slonim, R.L., Barron, G.: Predictive value and the usefulness of game theoretic models. Int. J. Forecast. **18**(3), 359–368 (2002)
26. Ert, E., Erev, I.: Replicated alternatives and the role of confusion, chasing, and regret in decisions from experience. J. Behav. Decis. Mak. **322**, 305–322 (2007)
27. Fudenberg, D., Levine, D.K.: The Theory of Learning in Games, MIT Press Books, vol. 1. The MIT Press (1998)
28. Fudenberg, D., Levine, D.K.: Conditional universal consistency. Games Econ. Behav. **29**(1–2), 104–130 (1999)
29. Ho, T.H., Camerer, C.F., Chong, J.K.: Self-tuning experience weighted attraction learning in games. J. Econ. Theory **133**(1), 177–198 (2007)
30. Hu, J., Wellman, M.P.: Nash q-learning for general-sum stochastic games. J. Mach. Learn. Res. **4**, 1039–1069 (2003)
31. Jafari, A., Greenwald, A.R., Gondek, D., Ercal, G.: On no-regret learning, fictitious play, and nash equilibrium: In: Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01, pp. 226–233. Morgan Kaufmann Publishers Inc., San Francisco, CA (2001)
32. Kearns, M., Littman, M.L., Singh, S.: Graphical models for game theory In: Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence (2001)
33. Kleinberg, R., Piliouras, G., Tardos, É.: Multiplicative updates outperform generic no-regret learning in congestion games In: Proceedings of the forty-first annual ACM symposium on Theory of computing (2009)
34. Koller, D., Milch, B.: Multi-agent influence diagrams for representing and solving games. Games Econ. Behav. **45**(1), 181–221 (2003)
35. Levitt, S.D., List, J.A.: Field experiments in economics: The past, the present, and the future. NBER Workingc Papers 14356 National Bureau of Economic Research Inc (2008)
36. Ng, B., Boakye, K., Meyers, C., Wang, A.: Bayes-adaptive interactive pomdps AAAI (2012)
37. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann Publishers Inc., San Francisco, CA (1988)
38. Rezek, I., Leslie, D.S., Reece, S., Roberts, S.J., Rogers, A., Dash, R.K., Jennings, N.R.: On similarities between inference in game theory and machine learning. J. Artif. Intell. Res. **33**(1), 259–283 (2008)
39. Shoham, Y., Powers, R., Grenager, T.: If multi-agent learning is the answer, what is the question? Artif. Intell. **171**(7), 365–377 (2007). doi:10.1016/j.artint.2006.02.006
40. Watkins, C.J.C.H.: Learning from delayed rewards. Ph.D. thesis, University of Cambridge (1989)

41. Waugh, K., Bagnell, D., Ziebart, B.D.: Computational rationalization: The inverse equilibrium problem. In: Proceedings of the 28th International Conference on Machine Learning, pp. 1169–1176, New York, NY (2011)
42. Wright, J.R., Leyton-Brown, K.: Behavioral game theoretic models: a bayesian framework for parameter analysis In; Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2 (2012)
43. Yang, R., Ordonez, F., Tambe, M.: Computing optimal strategy against quantal response in security games In: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2 (2012)