Applied Soft Computing 26 (2015) 368–377

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Evolutionary collective behavior decomposition model for time series data mining

Zengchang Qin [a], Tao Wan [b,*], Yingsai Dong [c], Yu Du [d]

[a] Intelligent Computing and Machine Learning Lab, School of ASEE, Beihang University, Beijing 100191, China
[b] School of Biological Science and Medical Engineering, Beihang University, Beijing 100191, China
[c] Department of Computer Science, University of British Columbia, Canada
[d] Courant Institute of Mathematical Sciences, New York University, USA

ARTICLE INFO

Article history:
Received 8 March 2013
Received in revised form 24 August 2014
Accepted 22 September 2014
Available online 7 October 2014

Keywords:
Minority games
Mixed games
Collective behavior decomposition
Genetic algorithms
Evolutionary mixed games learning

ABSTRACT

In this research, we propose a novel framework referred to as collective game behavior decomposition where complex collective behavior is assumed to be generated by aggregation of several groups of agents following different strategies and complexity emerges from collaboration and competition of individuals. The strategy of an agent is modeled by certain simple game theory models with limited information. Genetic algorithms are used to obtain the optimal collective behavior decomposition based on history data. The trained model can be used for collective behavior prediction. For modeling individual behavior, two simple games, the minority game and mixed game are investigated in experiments on the real-world stock prices and foreign-exchange rate. Experimental results are presented to show the effectiveness of the new proposed model.

1. Introduction

Collective intelligence is a shared or group intelligence that emerges from the collaboration and competition of many individuals and appears in consensus decision making of agents. Collective behaviors can be modeled by agent-based games in which each individual agent follows its own local strategies. Agent-based experimental games have attracted much attention in different research areas, such as psychology [21], economics [4,24] and financial market modeling [6,12,17]. Agent-based models (ABM) of complex adaptive systems (CAS) provide invaluable insight of the highly non-trivial collective behavior of a population of competing agents. Researchers aim to model these systems where the agents involved have similar capabilities, share global information and are competing for limited resources.

In a given complex system populated with a group of agents, it is unrealistic that the whole population follows the same strategy. The basic assumption is the existence of various types of strategies for agents. In this research, we propose a model that the behavior of an agent can be modeled using a simple game theory model. Aggregations of the individual behaviors become the collective behavior of the system. Two learning scenarios are considered in the proposed framework: Learning with *complete information* where all the data of agents' choices and behaviors in each round are available. However in reality, it is always infeasible to obtain all records of agents' choices in each round of the game. Since we have assumed that the collective data are generated from the combination of variant groups of agents' behaviors, how can we decompose the collective data into the combinations of micro-level data. That is referred to as learning with *incomplete information*.

In the proposed framework, the behavior of an agent is modeled by a game and can be determined by a set of parameters. A *genetic algorithm* (GA) can be used to optimize these parameters to get the best approximation of the original system. To further explore collective behaviors, we add variant groups of agents playing different game theory models and proposed a new data mining framework. We estimate the resource-constrained environment parameters to maximize the approximation of the system outputs and test the effectiveness of the proposed model in the real-world stock market. This framework provides a new way to understand the relationship between micro-behaviors and macro-behaviors in complex dynamic systems.

This paper is organized as follows: Section 2 introduces the recent related research. In Section 3, we describe the new model of collective behavior prediction with complete information. In Sections 4 and 5, we investigate the environment with incomplete

* Corresponding author. Tel.: +86 010 82316875; fax: +86 010 82316875.
E-mail addresses: zcqin@buaa.edu.cn, zengchang.qin@gmail.com (Z. Qin), tao.wan.wan@gmail.com (T. Wan), yingsaid@cs.ubc.ca (Y. Dong), ydu1989@gmail.com (Y. Du).

http://dx.doi.org/10.1016/j.asoc.2014.09.036
1568-4946/

information, we use GA to discover the composition of agent behaviors of the system based on the minority game and the mixed game, respectively. In Section 6, we apply the model to predict the real-world financial time series data to verify its effectiveness. Conclusions and discussions are given in the end.

2. Related work

In agent based modeling, repetitive and competitive interactions among agents generate complex behavioral patterns. The aggregation of simple interactions at the micro-level may generate sophisticated structures at the macro-level, providing valuable information about the dynamics of the real-world system [27]. In fact this intricate two-way feedback between micro-structure and macro-structure has been recognized within economics for a very long time [9,15,23]. One of the directions is agent-based financial market modeling [5]. The "bottom-up" models were proposed to use computational and mathematical tools to describe macro features emerging from a composition of individual interacting agents. For example, the Santa Fe Artificial Stock Market, one of the first agent-based financial market platforms born in the late 1980s, looks at financial markets from an agent-based perspective [14]. This virtual market originated from a desire for building a financial market with an ecology of trading strategies, i.e., successful strategies would persist and replicate, and weak strategies would go away. This project makes use of stochastic optimizations tools, such as the genetic algorithms and classifier system, to model the process of learning [10].

Minority game (MG) [3] is arguably the simplest model in agent-based modeling research. In such a game, an odd number (N) of agents successively compete to be in the minority, which can be regarded as a simplified version of *El Farol Bar Problem* [2] created in 1994 based on a bar in Santa Fe, New Mexico. It was one of the best known experimental game models in economics, in which a number of people decide weekly whether go to the El Farol bar to enjoy live music in the risk of staying in a crowd place or stay at home. Formally: N people decide independently to go or stay each week, they have two actions: go if they expect the attendance to be less than $\alpha N (0 < \alpha < 1)$ people or stay at home if they expect it will be overcrowded. There is no prior communication among the agents; the only information available is the numbers who came in past a few weeks. In the minority game, there are an odd number of players and each must choose one of two choices independently at each turn, players who end up on the minority side win. The minority game has the property that no single deterministic strategy may be adopted by all participants in equilibrium [3].

As a new tool for learning complex adaptive systems, the minority game has been applied to various areas especially in financial market modeling [12,13]. However, there are some weakness of using the basic MG model in real-world market data analysis. One critical weakness is that all agents have the same memory length so that the diversity of agents is limited. In order to cover such limitations, several MG variants were proposed:

(1) Grand Canonical Minority Game (GCMG): The GCMG [11] model is an extension of the classical MG, wherein an agent is rewarded for being in the minority, and it has the possibility not to trade, therefore allowing for a fluctuating agent number invested in the market.
(2) Grand Canonical Majority Game (GCMjG): In the GCMjG [20] model, an agent is rewarded for being in the majority instead of in the minority.
(3) Delayed Grand Canonical Majority Game (delGCMjG): In the delGCMjG [1] model, an agent is rewarded similarly to an agent in the GCMjG, but the return following the decision is delayed by one time step to reflect the more realistic market property.
(4) Delayed Grand Canonical Minority Game (delGCMG): The delGCMG [27] model is the analog of the delGCMjG, except for the minority payoff, whereby each agent is rewarded according to how the return at the next time step is compared with her decision taken at the previous time step. In other words, the delGCMG is a delayed GCMG.
(5) Mixed Game: In the real-world markets, some agents play the minority game, which are referred to as "foundation traders" who hope to maximize their profits; while others are just "trend chasers" who choose what the majority do (i.e., majority game). In order to establish an agent-based model which more closely approximate the real market, Gou [7] modifies the MG model by dividing agents into two groups: one group play the minority game and the other group play the majority game, thus this system is referred to as a mixed game model.

Extensive research in econophysics [19] has been done on agent-based experimental games from the perspective of interdisciplinary disciplines such as physics, mathematics and complexity science. For example, Sysi-Aho [25] proposed a genetic algorithm based adaptation mechanisms within the framework of the minority game, and found that the adaptation mechanism leads the market system fastest and nearest to maximum utility or efficiency. Gou [8] studied how the change of mixture of agents in the mixed game model can affect the change of average winnings of agents and local volatilities of the artificial stock market. Wang [26] proposed an extended minority game model called the market-directed resource allocation game (MDRAG) and investigated the influence of agents' decision-making capacity toward the efficiency, stability and predictability of the market system and its phase structure.

Unfortunately, fewer research focus on exploring macro-level collective behavior prediction by understanding the emergent properties of macro-level behavior from micro-level behavior. We can rarely see that agent-based models were put into practice of real market predictions, e.g., predicting fluctuation of the stock prices. In this paper, we assume that the collective data are generated from the combination of micro-behaviors of variant groups of agents employing different strategies. We then model and estimate the resource-constrained environment parameters to maximize the approximation of the system outputs to the real-world test data.

3. Agent-based minority game

3.1. Strategies of agents

Minority games [3] is one of the simplest game theory models. Suppose an odd number of N agents decide between two possible options, say to attend Room A or B in each round of the game. Formally, in round t ($t = 1, 2, \ldots, T$), each agent i takes an action $a_i(t)$ for $i = 1, 2, \ldots, N$ to choose between A and B, or formally:

$$a_i(t) = \begin{cases} A & \text{agent } i \text{ chooses room } A \\ B & \text{agent } i \text{ chooses room } B \end{cases} \quad (1)$$

At the t round of the game, agents belonging to the minority group win. The winning outcome can be represented by a binary function $w(t)$. If A is the minority side, i.e., the number of agents choosing Room A is no greater than $(N-1)/2$, we define the winning outcome $w(t) = 0$; otherwise, $w(t) = 1$. The winning outcomes

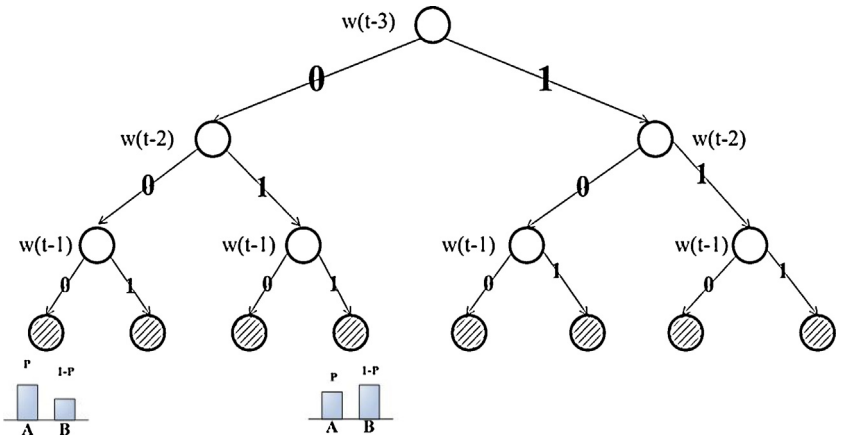

Fig. 1. A decision tree can be used to predict the next round outcomes based on estimated probability of going Room A or B.

to public are regarded as global information that can be formally represented by

$$w(t) = \begin{cases} 0 & \text{if: } \sum_{i=1}^{N} \Delta(a_i(t) = A) \le (N-1)/2 \\ 1 & \text{otherwise} \end{cases} \quad (2)$$

where $\Delta(\alpha)$ is the truth function:

$$\Delta(\alpha) = \begin{cases} 0 & \alpha \text{ is false} \\ 1 & \alpha \text{ is true} \end{cases} \quad (3)$$

We assume that agents make choices based on the most recent m winning outcomes $h(t)$, which is called *memory* and m is the *length of memory*.

$$h(t) = [w(t-m), \ldots, w(t-2), w(t-1)] \quad (4)$$

Given the outcome $w(t)$ at the time t, agent i keeps a record $r_i(t)$ that tells whether it has won the game or not.

$$r_i(t) = \begin{cases} Win & \text{Agent } i \text{ wins at time } t \\ Lose & \text{Agent } i \text{ loses at time } t \end{cases} \quad (5)$$

We usually assume that each agent's action toward the previous data is governed by a "strategy" [3]. The strategy is a lookup table based on the past m-bit memory which is described as a binary sequence, then there are 2^{2^m} possible strategies in the strategy space. Each agent looks into the most recent history for the same pattern of m bit string and predicts the outcome. Given the memory $h(t)$, the choice for the agent i is guided by the strategy S is denoted by $S(h(t))$. Table 1 shows one possible strategy with $m = 3$. For example, $h(t) = [0\,0\,1]$ represents that the agent who have chosen Room A win the game at $t-3$ and $t-2$, but lose at $t-1$. (see Eq. 2). The next round (at time t) choice for the current agent will be $S([0\,0\,1]) = B$. A strategy can be regarded as a particular set of decisions on the permutations of previous winning outcomes.

3.2. Decision tree learning for agents' strategies

For an agent, given the history of winning outcomes and his previous actions, we can use machine learning algorithm to learn its local strategy. First, we can sample the time series data with a sliding window into a training set. In round t, the target value (either A or B) is the agent's actual choice at the current round. Therefore, the training set for Agent i can be formally defined as

$$\mathcal{D}_i = \{(h(t), a_i(t))\} \text{ for } t = 1, 2, \ldots, T \quad (6)$$

Decision tree (DT) learning is one of the simplest and effective algorithms. DT and its variants have been widely used in numerous machine applications [18]. A decision tree can also be regarded as a set of rules. In this paper, we use the probabilistic estimation tree [16] where each branch of the tree is associated with a probability distribution.

Fig. 1 shows a decision tree for modeling behaviors of agent i based on the training data. For each branch $W = [w(t-m), \ldots, w(t-2), w(t-1)]$, there is an associated probability distribution on possible agent's choices (i.e., A or B) that is calculated based on the proportions of data falling through this branch. The reason for not using Naive Bayes is that it does not satisfy the independence assumption. On the opposite, it is fully dependent. This may result in poor performance in learning the strategy of an agent.

3.3. Learning with complete information

Previous studies on the MG model and other multi-agent systems usually assume agents are homogeneous [3,6,17]. Obviously, it is not a feasible assumption. We notice that, in real-life scenarios, the complexity of marketing world is embodied in existence of varieties types of agents using their own strategies. In the following experiment, N agents are divided into three groups. The agents in the first group make random choices between A and B following a uniform distribution, they are referred to as *random agents*. The second and the third group of agents follow two particular predefined strategies S_1 and S_2, respectively (e.g., see Table 2). Not only the history of winning outcomes, but also all the records of agent's choices are observable. Based on these data, we can use a decision tree to learn the patterns of other agents and make "smart" decisions.

Table 1
One sample strategy with $m = 3$, the current choice of agent is decided by its previous three steps memory.

$h(t)$	000	001	010	011	100	101	110	111
$S(h(t))$	A	B	B	A	B	A	B	B

Table 2
Two fixed strategies with $m = 4$ are used in experiments.

$h(t)$	0000	0001	0010	0011	0100	0101	0110	0111
$S_1(h(t))$	A	A	B	B	B	A	B	B
$S_2(h(t))$	A	B	B	B	A	A	A	B
$h(t)$	1000	1001	1010	1011	1100	1101	1110	1111
$S_1(h(t))$	B	A	A	A	A	A	B	B
$S_2(h(t))$	A	A	B	A	B	B	A	B

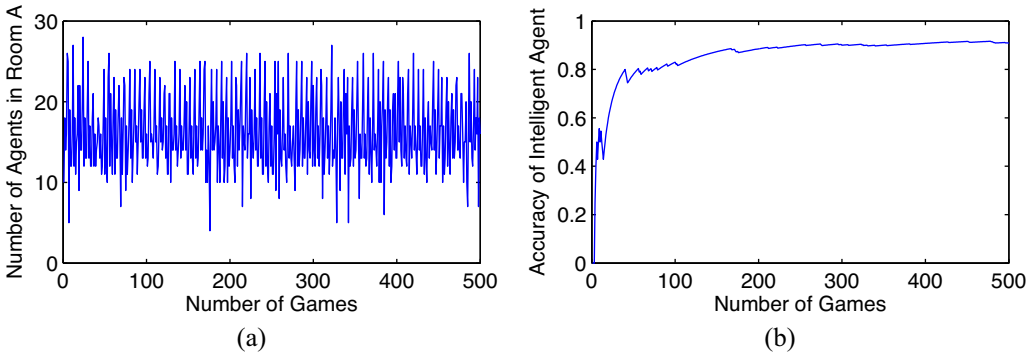

Fig. 2. (a) The number of agents in room A in 500 runs of the MG experiments with fixed strategies. (b) The accuracy of using a decision tree to predict the minority side.

For each agent i, its current choice $a_i(t)$ can be predicted by the decision tree learning based on the training data (see Eq. (6)). At time t, the probability of choosing A, $P_I(A)$, is calculated based on its estimation of other agents' choices $a_i(t)$ where $i = 1, \ldots, N$.

$$P_I(A) = 1 - \frac{\sum_{i=1}^{N} \Delta(a_i(t) = A)}{\sum_{i=1}^{N} \Delta(a_i(t) = A) + \Delta(a_i(t) = B)} \tag{7}$$

where $\Delta(\cdot)$ is defined by Eq. (3) and $P_I(B) = 1 - P_I(A)$. This can be interpreted that we should choose the room most of agents will not go in order to be the minority. Formally:

$$a_I(t) = \begin{cases} A & P_I(A) > P_I(B) \\ B & \text{otherwise} \end{cases} \tag{8}$$

The wining accuracy in T steps is evaluated by

$$AC_I(t) = \frac{\sum_{t=1}^{T} \Delta(r_I(t) = Win)}{\sum_{t=1}^{T} \Delta(r_I(t) = Win) + \Delta(r_I(t) = Lose)} \tag{9}$$

In the following experiment, we set the total number of agents $N = 31$, the number of random agents $N_r = 12$, the number of agents using S_1 and S_2 are $N_{s_1} = 6$ and $N_{s_2} = 12$, respectively. Fig. 2(b) shows the performance of using a decision tree for prediction in 500 rounds of games. The results show that the decision tree can take advantages by observing and learning from other agents' behaviors. We can predict which room is going to be the minority with accuracy around 90% after 500 runs. However, from the macro-level data shown in Fig. 2(a), the system still looks very random and unpredictable. The reason that we cannot achieve the 100% accuracy is because the injected uncertainties by random agents with no patterns.

From the above experimental results, we can conclude that a machine learning model such as a decision tree works almost perfectly with fixed strategies although over 30% of agents are unpredictable random agents. To study the influence of the random agents, we tested different percentage of random agents ranging from 0% to 100% and the winning accuracy is shown in Fig. 3. We can see that the accuracy of the decision tree is almost monotonically decreasing as the percentage of random agent increases. When the percentage of random agents becomes larger than 80%, the predictive accuracy becomes around 50%. The increasing dominance of random agents makes the system become totally random and unpredictable.

4. Behavior learning with genetic algorithms

In the previous section, we use a machine learning method to learn the patterns of agents with different strategies. The experimental results show that it can predict the minority side in different scenarios. It assumes the availability of complete information about the other agents, who went to which room in which round of the game. However, this is not a realistic assumption. In most cases, we can only obtain the collective data $w(t)$ but not the detailed agent behavior $r_i(t)$. This scenario is referred to as *incomplete information*. In this section, we aim to learn from the macro-level data $w(t)$ and propose a framework by assuming the macro-behavior can be decomposed into several groups of agents, within each group, agents employ similar strategies. A *genetic algorithm* [10] can be used to estimate the parameters of this decomposition in order to achieve the maximum likelihood. N agents are divided into a few groups. One group of agents are random agents, and other several groups of agents have fixed strategies. However, we have no idea how many agents in each group and which strategies this group of agents employ. We only know the history of winning outcomes $w(t)$ and an educated guessed maximum number of groups K. We use a vector of parameters to represent the number of agents in each group and the strategy they employ. The genetic algorithm

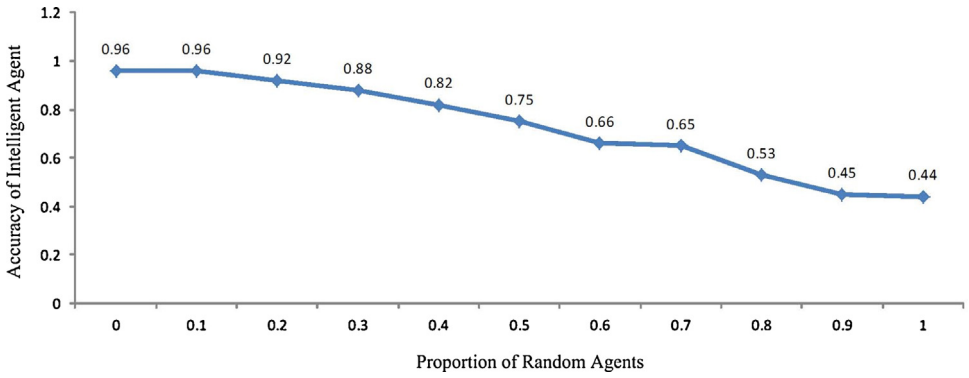

Fig. 3. Impact of random agents: results are obtained by using different percentage of random agents in 500 runs of the minority game.

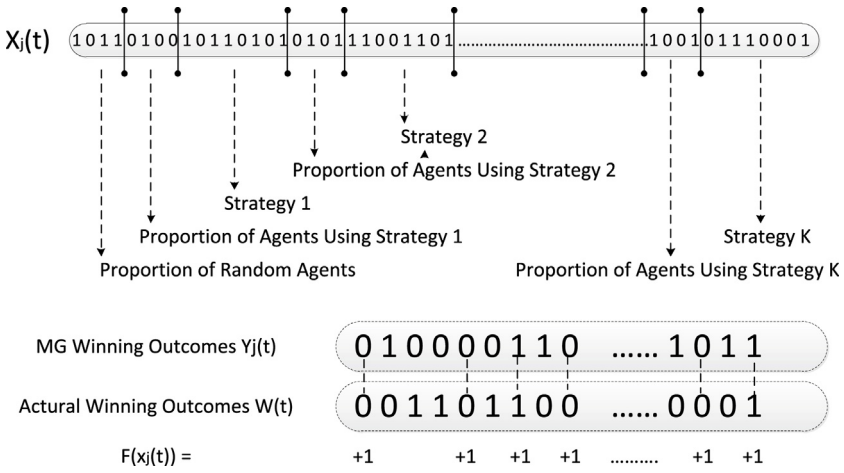

Fig. 4. The process for calculating the fitness function for a chromosome at time t. A chromosome is consisted by numbers of agents in each group and the strategies of this group. For each chromosome $\mathbf{x}_j$, we can obtain a sequence of winning outcomes $y_j(t)$ by running the MGs based on the given parameters. The fitness function is calculated based on the comparisons between $y_j(t)$ and the actual sequence of winning outcomes $w(t)$.

is used to optimize these parameters in order to obtain the most likely history of winning sequence.

4.1. Evolutionary collective behavior decomposition

In this experiment, we only use the information of winning outcomes $w(t)$ with a guessed number of groups with fixed strategies K. The agents can be divided into $K+1$ groups: $\{G_r, G_1, \ldots, G_K\}$, where group G_r is the group of random agents and G_k for $k = 1, \ldots, K$ employs the strategy S_k. We use the following parameters to define one MG: the percentage of random agents P_r, percentage of agents with one certain fixed strategy P_{S_k} where S_k is the strategy for the group. Therefore, a chromosome $\mathbf{x}$ is composed by the following parameters.

$$\mathbf{x} = \{P_r, P_{S_1}, S_1, \ldots, P_{S_K}, S_K\}$$

The process of fitness calculation is illustrated in Fig. 4. At time t of the game, in order to evaluate one chromosome $\mathbf{x}_j$ ($j = 1, \ldots, J$ where J is the population size in the GA), we run the MG with the parameter setting given by $\mathbf{x}_j$ to obtain the history of winning outcomes $y_j(t)$. Comparing $y(t)$ with the actual sequence $w(t)$: for t runs from 1 to a specified time T, once $y_j(t) = w(t)$, we add 1 to the fitness function $f(\mathbf{x}_j)$. Formally:

$$f(\mathbf{x}_j(t)) \leftarrow \begin{cases} f(\mathbf{x}_j(t)) + 1 & \text{if : } y_j(t) = w(t) \\ f(\mathbf{x}_j(t)) & \text{otherwise} \end{cases} \tag{10}$$

At each time t, the best chromosome $\mathbf{x}^*(t)$ is selected from the pool:

$$\mathbf{x}^*(t) = arg\max_j f(\mathbf{x}_j(t)) \text{ for } j = 1, \ldots, J$$

Given the best chromosome $\mathbf{x}^*(t)$, its parameters can give the best possible complete information scenario so that the intelligent agent can learn with a decision tree introduced in Section 3. This GA-based learning framework is referred to as evolutionary collective behavior decomposition model.

4.2. Simulated experiment

In the following experiments, we first simulate the real-world market by using initialized strategies selected from strategy space for each agent and generate a time series of macro-level data. Then we use the evolutionary collective behavior decomposition model to predict the macro-level time series based on the history data. We set up an artificial system with the following parameter settings: group number $K = 4$, strategies S_k are generated with memory length $m = 3$. Other parameters for the MG are: $N = 81$, $P_r = 0.04$, $P_{S_1} = 0.27$, $P_{S_2} = 0.31$, $P_{S_3} = 0.16$, $P_{S_4} = 0.21$. For the genetic algorithm: crossover rate $P_c = 0.7$, mutation rate $P_m = 0.01$ and population size $J = 50$.

Fig. 5(a) shows the time series data of how many agents went to room A in 1000 runs. It has the empirical mean nearly 41 and with a small variance. This property makes the prediction even harder because a small change of agent's choice may lead to a different winning outcome. The overall performance looks random and unpredictable. By using the method discussed in the previous section, we can obtain the results shown in Fig. 5(b). As we can see from the figure, as the generation increases, the accuracy of prediction increases monotonically before reaching a stabilizing value around 0.78, which is considerably high. That is also easy to understand that as in the given simulation environment, the macro-level time series were composed of micro-level behavior of agents employing strategies similar to that in Table 1.

The significance of these results is that, if the collective behavior of the system were generated from composition of micro-level behavior under these assumptions, by using genetic algorithm, we can effectively find the most likely combinations of individual behavior that could generate this macro-level sequence. Many real-world complex phenomena are considered to be related to the minority game [3,12,13], such as the fluctuations of stocks and currency exchange rates. Although the macro-level data are seemingly random and unpredictable, we can use this framework to represent the behavior compositions of the system. In other words, we are able to learn the collective behavior by decomposition of the games. We take the history of macro-level data as training set and estimate related micro behavior parameters to maximize the likelihood of the system Using such a composition model enables us to predict the future results from a macroscopic aspect.

5. Learning with mixed games

In order to obtain a better approximation of the collective behavior in the real-world market, Gou [7,8] modified the MG model and proposed the "mixed game model", in which the fixed

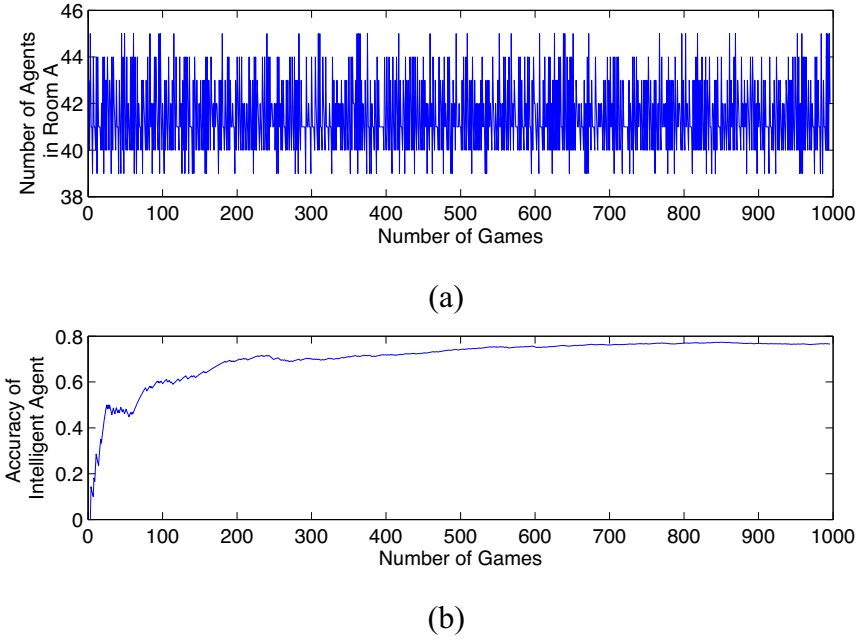

Fig. 5. (a) Macro-level data: the number of agents in room *A*. (b) Accuracy of the intelligent agent by using genetic algorithm.

strategies of agents are divided into two groups: group G_N plays minority game with the same strategy, while Group G_J plays majority game with the same strategy. Comparing to the MG model, the most significant part of mixed game is that it has an additional group of "trend chasers", therefore it is more realistic to simulate a real-world market.

In the training process, all agents in G_N choose the best strategy with which they can predict the minority side most accurately, while all agents in G_J choose the best strategy with which they can predict the majority side most accurately. N_1 represents the number of agents in G_N and N_2 represents the number of agents in G_J. We use m_1 and m_2, respectively, to describe the memory lengths of these two groups of agents. As each agent's reaction is based on a strategy corresponding a response to past memories, there are $2^{2^{(m_1)}}$ and $2^{2^{(m_2)}}$ possible strategies for G_N or G_J, respectively, which composite the whole strategy spaces.

However, it is unrealistic to assume all agents playing only the minority game or the majority game, we assume the existence of random agents as well. Then we have 3 groups of agents as the following.

- Group G_N: agents who play minority game.
- Group G_J: agents who play majority game.
- Group G_R: agents who make random decisions.

For G_N and G_J we assume that the overall effect can be decomposed into several small subgroups, while each subgroup of agents uses a certain strategy. The decomposition of the collective behavior involves a big set of parameters including the agent number in each subgroup and the strategies they employ. We then use genetic algorithms to tune these parameters to yield the collective behavior with best approximation of the history data.

5.1. Chromosome encoding for models with mixed games

In the new model, we use a parameter vector to represent the number of agents of each subgroup and the corresponding strategy they use. Given the history winning outcomes $w(t)$, the expected maximum number of subgroups using fixed strategies in G_N is K_N, and the expected maximum number of subgroups using fixed strategies in G_J is K_J. Thus agents can be divided into $K_N + K_J + 1$ groups:

$$\{G_R, G(S_N^1), \ldots, G(S_N^{K_N}), G(S_J^1), \ldots, G(S_J^{K_J})\}$$

where G_R represents the group of random agents, $G(S_N^i)$ (for $i = 1, \ldots, K_N$) represents the subgroup of agents holding strategy S_N^i in Group G_N. $G(S_J^k)$ (for $k = 1, \ldots, K_J$) represents the subgroup agents holding strategy S_J^k in Group G_J. Therefore, the chromosome $\mathbf{x}$ is encoded in the following form:

$$\mathbf{x} = \{P_R, P(S_N^1), S_N^1, \ldots, P(S_N^{K_N}), S_N^{K_N}, P(S_J^1), S_J^1, \ldots, P(S_J^{K_J}), S_J^{K_J}\}$$

- P_R: the percentage of random agents among all agents (i.e., $P_R = |G_R|/N$).
- $P(S_N^i)$: the percentage of the number of agents in the minority game subgroup i ($i \in [1, 2, \ldots, K_N]$) with the fixed strategy S_N^i (i.e., $P(S_N^i) = |G(S_N^i)|/N$).
- S_N^i: binary coding of the minority game strategy S_N^i.
- $P(S_J^k)$: the percentage of the number of agents in the majority game subgroup k ($k \in [1, 2, \ldots, K_J]$) with the fixed strategy S_J^k (i.e., $P(S_J^k) = |G(S_J^k)|/N$).
- S_J^k: binary coding of the majority game strategy S_J^k.

Fig. 6 illustrates that the collective behavior is a combination of choices from the above three types of agents. Given history data $h(t)$, we can use GA to explore all possible combinations of subgroups and compositions of the market, then use this information to make smart choices.

5.2. Fitness function

By using genetic algorithms, we aim to generate a system in which agents from both Group G_N and Group G_J can achieve their

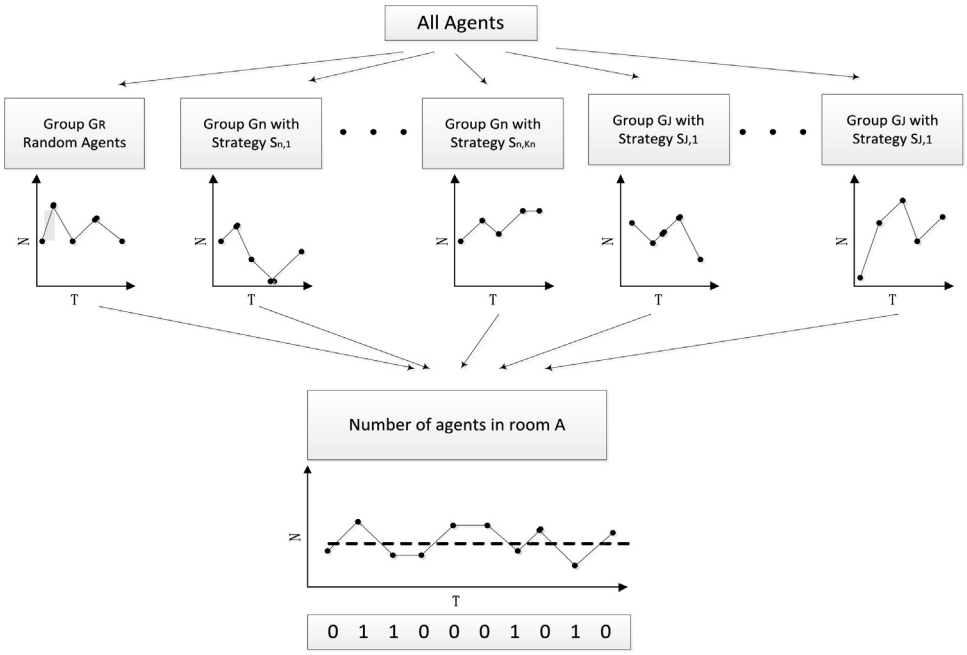

Fig. 6. The generative process of collective data. All agents are divided into $K_N + K_J + 1$ groups where agents in the same subgroups act identically based on one particular strategy. The collective behavior can be regarded as an aggregation of all agents' behavior.

goals to the greatest extent, i.e., agents in Group G_N end up on the minority side while agents in Group G_J end up on the majority side. The final goal of the model is to obtain the best prediction of the market and make rational choice to maximize their profits. In round t, in order to evaluate the chromosome $\mathbf{x}_j (j = 1, 2, \ldots, J$ where J is the population size), we run the mixed game with parameter setting decoded from $\mathbf{x}_j$ and obtain the prediction. We choose the best chromosome by calculating the fitness function $f(\mathbf{x}_j)$ with the following three rules.

In round t, we consider collective data within the previous T steps: $(t-1-T, t-T, , \ldots, t-2, t-1)$

- *Rule 1*: For all agents in Group G_N, every time an agent predicts the correct outcome, i.e., chooses on the minority side, we add 1 to $f(\mathbf{x}_j)$.
- *Rule 2*: For all agents in Group G_J, every time an agent predicts the correct outcome, i.e., chooses on the majority side, we add 1 to $f(\mathbf{x}_j)$.
- *Rule 3*: If the prediction outcome $y_i(t)$ by the model is equal to the real-world macro outcome $w(t)$, we add a specific weight W_p to $f(\mathbf{x}_j)$.

Usually we set the weight value by a specific percentage of the total number of agents N:

$$W_p = \beta N \quad (\beta \in [0, 1]) \tag{11}$$

We calculate the fitness function $f(\mathbf{x}_j)$ for $t_0 = t-1-T, t-T, , \ldots, t-2, t-1$ and select the best chromosome $\mathbf{x}_j^*$ within the time range T.

$$\mathbf{x}^*(t) = arg\max_j f(\mathbf{x}_j(t)) \; for \; j = 1, \ldots, J \tag{12}$$

Then we decode parameters from the best chromosome and get the information of the combination, and then therefore predict the dynamics of the system.

6. Experiments on real-world market data

This novel framework of collective behavior decomposition paves a new way of using game theory models and evolutionary optimization to investigate the relationship between micro-level and macro-level data. Given a sequence of macro-level market data, we can use GA to explore the most likely combinations of single behavior that could generate this sequence. Many real-world complex phenomena are caused by aggregations of agents' behavior such as stock market and currency exchange rate, which are regarded as complicated and unpredictable in classical economics. In the following experiments, we apply this new proposed model to explore the compositions of stock market data using agents playing only the minority game and the mixed games.

6.1. Experimental data

In the following experiments, the new proposed model is tested on the real-world time series data. Two variants are considered: In the MGDM (Minority Game Data Mining) model, agent behavior is modeled by the minority game while agent behavior in the EMGL (Evolutionary Mixed Game Learning) is by the mixed-game. We randomly select 15 stocks from the Chinese stock market, data are collected from the China Merchants Securities,[1] and also the US Dollar-RMB (Chinese Renminbi) exchange rate.[2] The index and stock names are listed in Table 3.

Given opening price V_b and the closing price V_f for one trading day t, fluctuation of the market can be represented by winning outcomes in game $w(t)$. If $V_b > V_f$, then $w(t) = 1$; otherwise, $w(t) = 0$. One trading day can be regarded as one round of the game. By correctly predicting $w(t)$ using the learning model, we can capture the ups and downs of the market prices. We use the winning outcomes from certain 500 trading days as training data to learn the best

[1] Website: http://big5.newone.com.cn/download/new_zszq.exe.
[2] Data can be obtained from: http://bbs.jjxj.org/thread-69632-1-7.html.

Table 3
Real-world time series data from Chinese stock markets.

Index	Stock	Index	Stock
600036	China Merchants Bank Co.	600690	Qingdao Haier Co.
600000	SPD Bank Co.	600619	Baoshan Iron Steel Co.
600900	China Yangze Power Co.	600016	Minsheng Banking Co.
600011	Huaneng Power Indus. Inc.	600028	China Petro. &Chem. Co.
600038	Hafei Aviation Ind. Co.	600050	China United Net. Comm.
600056	CNTIC Trading Co.	600060	Hisense Electric Co.
600088	China Television Ltd.	600115	China Eastern Airlines Co.
600966	Shangdong Bohui Paper		

parameter setting, and then use it to predict the results in next 300 trading days. We compare the results of using the minority game model, the mixed game model, and the random guess in order to test performance.

In the following experiments, we set $K_N = K_J = 20$. Since almost all agents play with history memories of 6 or less in a typical MG [22], and m_N is usually larger than m_J when using mixed game model to simulate real market [7], we set $m_N = 3, 4, 5, 6$ and $m_J = 3$ to establish four configurations for the mixed game model. We set $K = 20$ and $m = 3$ in the minority game model. As for the GA, we set the population size $J = 50$, crossover rate $P_c = 0.8$, mutation rate $P_m = 0.05$, the specific weight $\beta = 0.5$. We run the whole experiments for 30 times to reduce the influences of randomness from GAs.

6.2. Data analysis

Table 4 shows the prediction accuracy of 15 stocks and the USD-RMB exchange rate within 300 trading days, where EMGL (6-3) represents the behavior decomposition model using mixed games with $m_N = 6$, $m_J = 3$. We use different configurations of memory length ($m_N = 3, 4, 5, 6$; $m_J = 3$) and calculate the mean prediction accuracy and its corresponding standard deviations. For most of the cases, the evolutionary behavior decomposition models perform obviously better than the random walk. And also the mixed model performs slightly better than the model with minority game only (for 9 of 15 stocks and the exchange rate, whose index numbers are 1, 3, 6, 9, 12, 13, 14, 15, 16). By adding agents who play majority game, we can generate a more realistic market and predict the stock prices more accurately. We can see that the accuracies with mixed game model are similar with the minority model on some stocks.

By observing the experimental results, we find that the prediction accuracy of USD-RMB exchange rate is much higher than the accuracy of stocks. In Fig. 7, we can see both the EMGL model and the MGDM model can predict with high accuracy (the mean accuracy is up to 58.6% for MGDM and 64.13% for EMGL (5-3)). We believe that this indicates a strong existing pattern captured by our models. The pattern may result from the Sino-US currency policies, which could be influenced more by centralized power rather than unorganized trading individuals. In this case, our model could be a promising method to test how much a market is affected, whether it is led by only several influential organizations or by equivalent individuals. Of course this still needs more future investigations.

Figs. 8–10 show the performances on stock # 14, # 2, *and* # 4, which are three representational results in our experiments. The first kind of results is like that on stock # 14 shown in Fig. 8, the EMGL model outperforms the MGDM model and both models outperform the random guess. 9 of 16 experimental data have similar performance. The second kind of results is like that on stock # 2 shown in Fig. 9. The EMGL model and the MGDM model have similar accuracy (which are not statistically different from each other), the two prediction curves are overlapped in the last, and both models outperform the random guess. So we are not able to tell which model is statistically better. The third kind of results is like that on stock # 4 shown in Fig. 10. The stock # 4 is the only stock in our experiment who has this results, that the MGDM model outperforms the EMGL model which means that the minority game modeling could be more appropriate than mixed games in this case.

The stock prices are driven by complex behavior and influenced by many unknown factors, it is hard to tell what sort of micro-behavior could be more appropriate than others. However, empirical results on these data have shown that the proposed learning framework of collective data decomposition is effective in solving this difficult problem. Though the EMGL model performs statistically better than the MGDM model for most of the cases in our experiments, we still need to be cautious about choosing between the MGDM model and the EMGL model (as well as different configurations of memory lengths), the performance of these two models may vary with specific stocks when making predictions of the market. The computation time of 30 rounds of GAs on

Table 4
Comparisons of mean accuracy and standard deviations of the EMGL and MGDM models on 16 real-world financial time series data. Data of 500 trading days are used for training and the next 300 trading days are for test. 15 stocks from Chinese stock market and the USD-RMB exchange rate are tested.

Stock index	MGDM	EMGL (3-3)	EMGL (4-3)	EMGL (5-3)	EMGL (6-3)
FX	58.59 ± 3.60	63.43 ± 2.36	63.78 ± 2.86	$\mathbf{64.13 \pm 2.71}$	62.51 ± 2.88
600036	$\mathbf{53.60 \pm 2.15}$	52.38 ± 2.10	53.08 ± 2.11	52.53 ± 2.11	53.03 ± 1.87
600690	50.81 ± 1.97	54.26 ± 1.15	54.01 ± 1.55	$\mathbf{54.56 \pm 1.71}$	54.02 ± 1.44
600000	$\mathbf{55.99 \pm 3.34}$	51.23 ± 2.80	52.41 ± 2.36	49.17 ± 2.31	51.63 ± 2.32
600019	$\mathbf{52.52 \pm 2.47}$	50.16 ± 5.10	45.86 ± 3.80	50.93 ± 3.53	52.22 ± 3.38
600966	53.94 ± 2.82	55.25 ± 2.59	54.45 ± 2.83	$\mathbf{56.22 \pm 3.01}$	54.54 ± 2.75
600900	53.49 ± 2.63	$\mathbf{53.71 \pm 2.15}$	48.48 ± 2.73	52.79 ± 2.00	52.45 ± 2.87
600016	$\mathbf{55.71 \pm 3.00}$	54.49 ± 2.75	52.63 ± 2.33	54.66 ± 1.93	52.44 ± 1.96
600011	50.87 ± 2.80	$\mathbf{51.49 \pm 1.46}$	51.21 ± 1.46	51.62 ± 1.56	51.21 ± 1.74
600028	$\mathbf{52.52 \pm 2.49}$	50.30 ± 2.75	51.62 ± 1.23	51.77 ± 0.83	51.41 ± 2.13
600038	53.14 ± 1.92	53.16 ± 2.18	52.85 ± 1.57	52.60 ± 2.04	$\mathbf{53.62 \pm 1.62}$
600050	51.35 ± 3.29	54.06 ± 2.02	54.38 ± 1.63	53.83 ± 1.05	$\mathbf{54.59 \pm 0.85}$
600056	52.99 ± 1.85	53.91 ± 3.03	54.84 ± 2.69	$\mathbf{55.09 \pm 2.66}$	54.53 ± 2.70
600060	53.13 ± 3.17	56.93 ± 1.58	56.92 ± 1.89	56.79 ± 1.74	$\mathbf{57.68 \pm 1.74}$
600088	50.69 ± 2.48	51.76 ± 3.12	52.11 ± 2.96	$\mathbf{54.14 \pm 2.18}$	53.29 ± 1.79
600115	55.62 ± 2.04	57.13 ± 0.92	$\mathbf{57.14 \pm 0.84}$	56.69 ± 1.08	56.44 ± 1.57

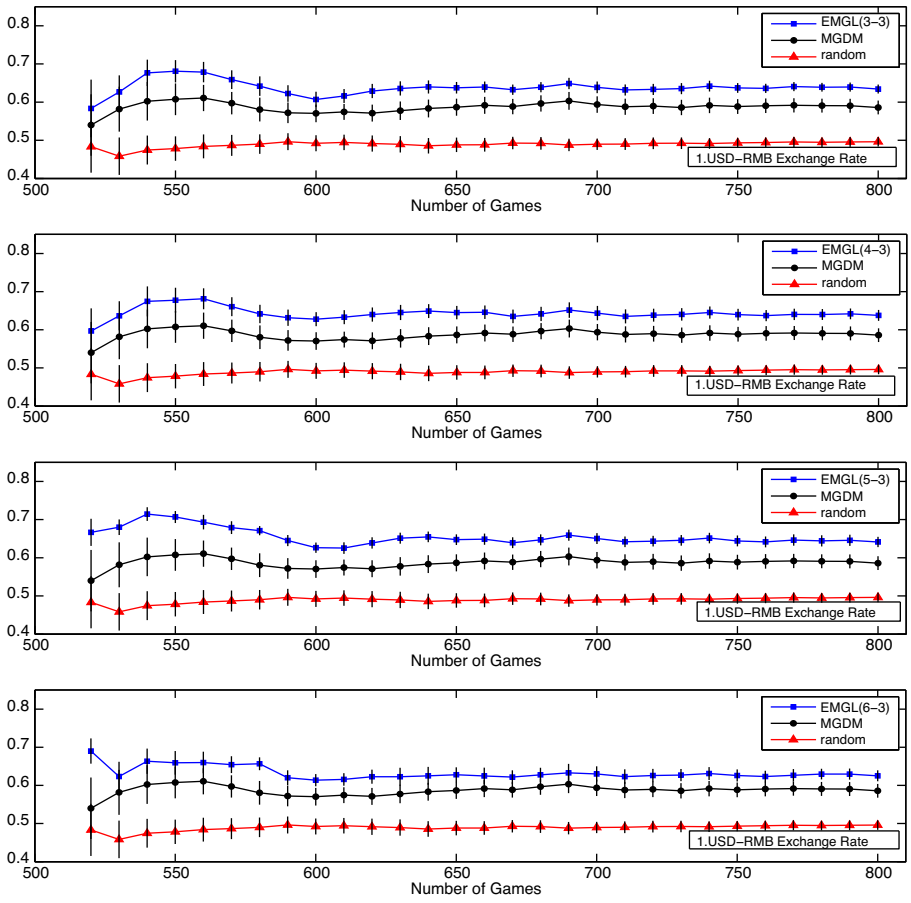

Fig. 7. Performance of the MGDM model and the EMGL model with different memory lengths on the USD-RMB exchange rate.

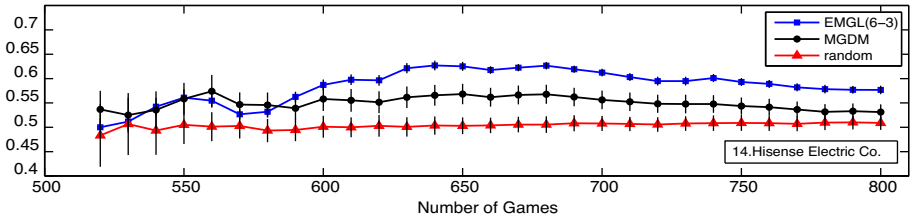

Fig. 8. Performance of the MGDM model and the EMGL model on stock # 14.

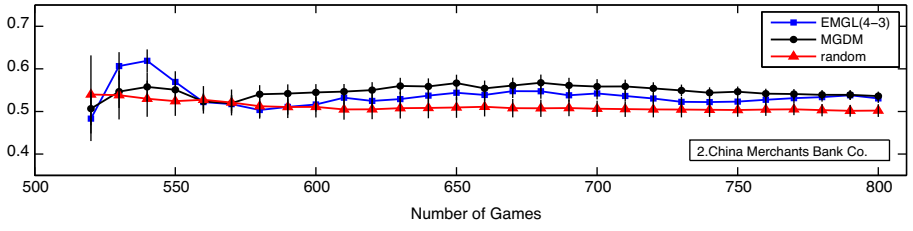

Fig. 9. Performance of the MGDM model and the EMGL model on stock # 2.

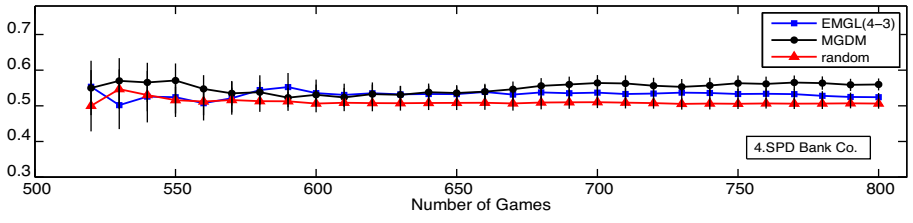

Fig. 10. Performance of the MGDM model and the EMGL model on stock # 4.

the given 12 datasets is about 5 h running the Matlab code on an Intel Pentium dual-core PC.

7. Conclusions and future work

In this paper, we proposed a novel learning framework in which the collective market behavior is considered as an aggregation of behavior of subgroups. By using genetic algorithms to explore all the possibilities of decomposition of the system, the proposed framework has the ability to capture the possible combinations based on the history data, and these combinations should have the maximum likelihood to the real structure of the system. We tested the model based on the minority game (MGDM) and the mixed game (EMGL) on a few real-world stock data and the USD-RMB exchange rate. We found that the new proposed framework consistently outperforms the random guesses. We can capture some week trends in the stock data which has been regarded as unpredictable random walk in classical economics. The EMGL performs a little better in the experiments comparing to the MGDM model, the possible reason is because the mixed game model is a more realistic approximation of the real-world market. However, we still need test on more data and further investigation to verify effectiveness of this new proposed model. Another future work is to apply this framework to other areas rather than financial data which are highly random and much corrupted by noise.

Acknowledgment

This work is supported by the National Science Foundation of China Nos. 61305047 and 61401012.

References

[1] J. Andersen, D. Sornette, The $-game, Eur. Phys. J. B: Condens. Matter Complex Syst. 31 (1) (2003) 141–145.
[2] W. Arthur, Bounded rationality and inductive behavior, Am. Econ. Rev. 84 (1994).
[3] D. Challet, Y. Zhang, Emergence of cooperation in an evolutionary game, Physica A 246 (1997).
[4] J. Doyne Farmer, Duncan, Foley, The economy needs agent-based modelling, Nature 460 (2009) 685–686.
[5] J.D. Farmer, D. Foley, The economy needs agent-based modelling, Nature 460 (2009) 685–686.
[6] D. Gode, S. Sunder, Allocative efficiency of markets with zero-intelligence traders: market as a partial substitute for individual rationality, J. Polit. Econ. 101 (1) (1993) 119–137.
[7] C. Gou, Dynamic behaviors of mix-game model and its application, Chin. Phys. 15 (6) (2006) 1239 http://arxiv.org/abs/physics/0504001
[8] C. Gou, Agents play mix-game, in: Econophysics of Stock and Other Markets, LNCS, Part II, 2006, pp. 123–132.
[9] F. Hayek, Individualism and Economic Order, University of Chicago Press, Chicago, 1948.
[10] J. Holland, Emergence: From Chaos to Order, 1998.
[11] N. Johnson, M. Hart, P. Hui, D. Zheng, Trader dynamics in a model market, J. Theor. Appl. Finance 3 (2000) 443–450.
[12] N. Johnson, P. Jefferies, P. Hui, Financial Market Complexity, Oxford University Press, Oxford, 2003.
[13] T. Lo, P. Hui, N. Johnson, Theory of the evolutionary minority game, Phys. Rev. E 62 (2000).
[14] B. LeBaron, Building the Santa Fe Artificial Stock Market, Physica A, 2002.
[15] M. Olsen, The Logic of Collective Action, Harvard University Press, Cambridge, MA, 1965.
[16] Z. Qin, Naive Bayes classification given probability estimation trees, in: Proceedings of Fifth International Conference on Machine Learning and Applications (ICMLA-2006), 2006, pp. 34–39.
[17] Z. Qin, Market mechanism designs with heterogeneous trading agents, in: Proceedings of Fifth International Conference on Machine Learning and Applications (ICMLA-2006), 2006, pp. 69–74.
[18] Z. Qin, J. Lawry, Decision tree learning with fuzzy labels, Inf. Sci. 172 (1–2) (2005) 91–129.
[19] R. Mantegna, H. Stanley, An Introduction to Econophysics: Correlations and Complexity in Finance, Cambridge University Press, 1999.
[20] M. Marsili, Market mechanism and expectations in minority and majority games, Physica A 299 (1–2) (2001) 93–103.
[21] A. Rapoport, A. Chammah, C. Orwant, Prisoner's Dilemma: A Study in Conflict and Cooperation, University of Michigan Press, Ann Arbor, 1965.
[22] R. Savit, K. Koelle, W. Treynor, R. Gonzalez, in: K. Tumer, D.H. Wolpert (Eds.), Collectives and the Design of Complex System, Springer-Verlag, 2004, pp. 199–212.
[23] A. Smith, An inquiry into the nature and causes of the wealth of nations, in: E. Cannan (Ed.), American Modern Library Series, The Modern Library, New York, 1937.
[24] V. Smith, An experimental study of competitive market behavior, J. Polit. Econ. 70 (1962) 111–137.
[25] M. Sysi-Aho, A. Chakraborti, K. Kaski, Searching for good strategies in adaptive minority games, Phys. Rev. E 69 (2004).
[26] W. Wang, Y. Chen, J. Huang, Heterogeneous preferences decision-making capacity and phase transitions in a complex adaptive system, Proc. Natl. Acad. Sci. U. S. A. 106 (21) (2009) 8423–8428.
[27] J. Wiesinger, D. Sornette, J. Satinover, Reverse engineering financial markets with majority and minority game using genetic algorithms, Swiss Finance Inst. Res. Pap. 10 (08) (2010).