Applied Soft Computing 36 (2015) 357–367

Contents lists available at ScienceDirect

Applied Soft Computing

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

# Ensemble learning of rule-based evolutionary algorithm using multi-layer perceptron for supporting decisions in stock trading problems

Shingo Mabu*, Masanao Obayashi, Takashi Kuremoto

*Graduate School of Science and Engineering, Yamaguchi University, Tokiwadai 2-16-1, Ube, Yamaguchi 755-8611, Japan*

ARTICLE INFO

*Article history:*
Received 18 July 2014
Received in revised form 30 June 2015
Accepted 7 July 2015
Available online 30 July 2015

*Keywords:*
Ensemble learning
Evolutionary computation
Rule-based algorithm
Multi-layer perceptron
Trend judgment
Stock trading

ABSTRACT

Classification is a major research field in pattern recognition and many methods have been proposed to enhance the generalization ability of classification. Ensemble learning is one of the methods which enhance the classification ability by creating several classifiers and making decisions by combining their classification results. On the other hand, when we consider stock trading problems, trends of the markets are very important to decide to buy and sell stocks. In this case, the combinations of trading rules that can adapt to various kinds of trends are effective to judge the good timing of buying and selling. Therefore, in this paper, to enhance the performance of the stock trading system, ensemble learning mechanism of rule-based evolutionary algorithm using multi-layer perceptron (MLP) is proposed, where several rule pools for stock trading are created by rule-based evolutionary algorithm, and effective rule pools are adaptively selected by MLP and the selected rule pools cooperatively make decisions of stock trading. In the simulations, it is clarified that the proposed method shows higher profits or lower losses than the method without ensemble learning and buy&hold.



## 1. Introduction

Stock markets are quite complicated real world environments where a huge number of factors influences the movement of the markets. Basically, there are two methods to predict the trends of markets and decide the timing of buying and selling stocks, i.e., fundamental analysis and technical analysis. Fundamental analysis is based on the financial statement reported by companies, economic trends of domestic and international environments, international relationships, and so on. Technical analysis is based on numerical analysis using the past movement of the stock prices, where many technical indexes such as moving average, golden cross, dead cross, etc. are used for stock trading decisions. Stock trading models and stock price prediction based on computational intelligence basically belong to technical analysis, and many methods using softcomputing and machine learning algorithms such as neural networks [1], genetic algorithms [2], genetic programming [3], support vector machines [4] have been proposed. The proposed method in this paper also belongs to technical analysis.

Due to the high complexity of stock markets, it is very difficult for single centralized system to represent and predict the movements of the markets because a huge number of rules covering the possible situations must be contained. Divide and conquer is a famous concept to solve a complicated problem by dividing it to several small problems [5]. Several expert programs for each small problem cooperatively solve their target problems, which contributes to making the whole system be compact. In the research of pattern recognition, many methods based on divide and conquer have been proposed to enhance the generalization ability of classification. One of the famous mechanisms is ensemble learning, where several classifiers are created and the classification is made by combining the results generated by the classifiers. Bootstrap aggregating (Bagging) [6] is one of the ensemble learning methods which creates several classifiers using bootstrapping samples [7], and makes classification by a majority vote. Adaptive boosting (AdaBoost) [8] creates several classifiers one by one considering the learning results of the previous classifiers. AdaBoost gives large weights on misclassified data and small weights on correctly classified data, therefore, the classifiers created later phase will execute learning concentrating on the misclassified data. Random forests [9] enhances Bagging algorithm and creates several decision trees by randomly selecting the fixed number of attributes for learning. In this paper, to apply the concept of ensemble learning to a stock

* Corresponding author. Tel.: +81 836 85 9519; fax: +81 836 85 9519.
*E-mail address:* mabu@yamaguchi-u.ac.jp (S. Mabu).

http://dx.doi.org/10.1016/j.asoc.2015.07.020
1568-4946/

trading model, an algorithm that selects good combinations of classifiers by multi-layer perceptron (MLP) considering the trends of stock markets is proposed.

In the proposed method, a large number of stock trading rules combining technical indexes are extracted by a graph based evolutionary algorithm named genetic network programming (GNP) [10,11]. GNP has been proposed to create programs mainly for dynamic environments using directed graph structures. Directed graph structures have some advantages comparing to tree structures used in genetic programming (GP) [12], for example, compact structures, repetitive process generation, and adaptability to non-Markovian process. GNP has three basic components: start node, judgment node and processing node, and combines them to make many if-then rules. There have been studies on knowledge extraction from data and representing the extracted knowledge as if–then rules. For example, in [13], a knowledge extraction method using probabilistic and fuzzy logical reasoning is proposed to generate if–then rules, and in [14], if–then rule extraction method from noisy time series data is proposed. The unique points of the proposed method in this paper are the if–then rule extraction utilizing the distinguished representation abilities of graph structures and its extension using MLP for realizing effective ensemble learning.

GNP has been applied to many applications such as elevator group supervisory control systems [15], data mining [16,17], network intrusion detection systems [18], stock trading models [19]. The algorithm used in this paper is a rule-based GNP [20] which extracts a large number of rules through the evolutionary process. General evolutionary computation basically aims to evolve individuals, and the best individuals obtained in the last generation become the solutions to the problems. However, rule-based GNP extracts rules from the elite individuals every generation, therefore, the rule extraction is carried out by all the elite individuals throughout the generations. This is the different point from the general framework of evolutionary computation.

In the rule extraction using GNP, the training period is divided into several sub-periods, and the expert rules for each sub-period are generated separately, which results in making several rule pools, i.e., classifiers, for ensemble learning. Then, MLP is used to calculate the similarity degrees of the stock data in the testing phase to the sub-periods in the training phase. In summary, MLP effectively selects good rule pools for stock trading in the current situation according to the similarity/matching calculation between training and testing data. The combination of rule-based decision making model, where a large number of rules are contained for adapting to various kinds of situations, and the automatic rule pool (classifier) selection by MLP can effectively make trading decisions.

There have been many studies on financial trading systems using artificial neural networks, evolutionary computation, optimization algorithms and so on. In [21], particle swarm optimization (PSO) is applied to the decision making in stock trading problems. PSO optimizes the weights on several technical indexes using multi-objective evaluation functions. In [22], an artificial neural network (ANN) is applied to a model that predicts future trading signals. In addition, a dynamic threshold method is proposed to judge whether the trading signals generated by ANN satisfy the conditions of buying or selling decisions. In [23], gene expression programming (GEP) is applied to the trading on mutual funds. GEP is an evolutionary algorithm which extends GP by using linear gene structures, and can explicitly represent trading rules by evolved tree structures. Comparing to the above methods, there are different features in the proposed method. First, the proposed method extracts a large number of association rules and stores them in the rule pools, therefore, various kinds of knowledge on the stock market movements can be saved for decision making. Both the proposed method and GEP [23] can explicitly represent trading rules, however, the proposed method is a rule-based system, while GEP is a tree structure-based system. ANN model [22] shows distinguished generalization ability to predict trading signals, but it is a black box model, thus it is difficult to explicitly show the rules of buying and selling decisions. Second, the proposed method uses only the important rules for the decision in the current situation by introducing an ensemble learning with MLP. In [21], PSO determines the importance of each technical index by adjusting weights, but it does not adapt to the changes of the trends in stock markets.

This paper is organized as follows. Section 2 reviews the rule extraction algorithm using rule-based GNP. Section 3 explains the rule pool generation and how to make MLP for selecting good rule pools depending on the stock market conditions. Section 4 introduces the simulation environments of the stock trading and analyzes the simulation results, and finally Section 5 is devoted to conclusions.

## 2. Review of rule-based genetic network programming

GNP has been proposed as an extension of GP in terms of the gene structure [11]. The gene structure of GNP is represented by a directed graph, which shows some advantages over tree structures of GP such as reusability of nodes, compact structure and decision making considering past events. In this section, the representation of the programs, how to execute and evolve the programs, and how to make rule pools are explained. More detailed explanation is described in [19].

### 2.1. Basic structure

The basic program structure is shown in Fig. 1, where there are three major components: start node, judgment node and processing node. Each judgment node has a if–then branch decision function and each processing node has an action function. In the stock trading problem, for example, the judgment nodes in Fig. 1 examine technical indexes generally used in the technical analysis and select one branch to the next node according to the judgment result such as {A = high, B = low}. The processing nodes determine buying and selling actions. The role of the start node is to determine the first node to be executed. Therefore, the node transition starts from the start node, and judgment and processing nodes are sequentially executed according to the connections between nodes. If the connections are optimized by the evolutionary algorithm, good action rules can be obtained.

In this paper, an extended structure of GNP is used to make programs, that is, GNP with reinforcement learning (GNP-RL) [11]. The feature of GNP-RL is that there are several subnodes in each judgment and processing node (Fig. 2). Each subnode has its own function and *Q*-value [24], and in the node transition, reinforcement

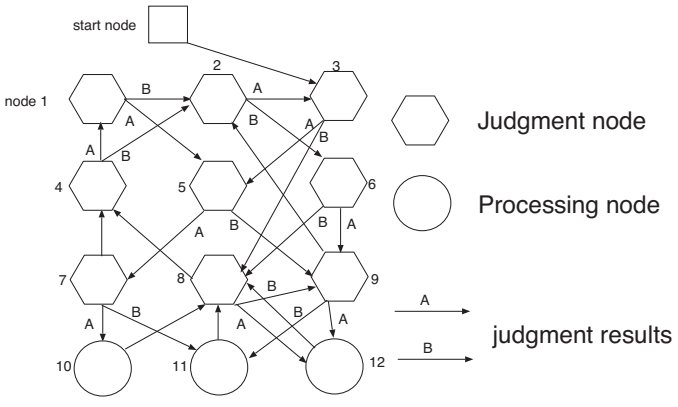



**Fig. 1.** Basic structure of GNP.

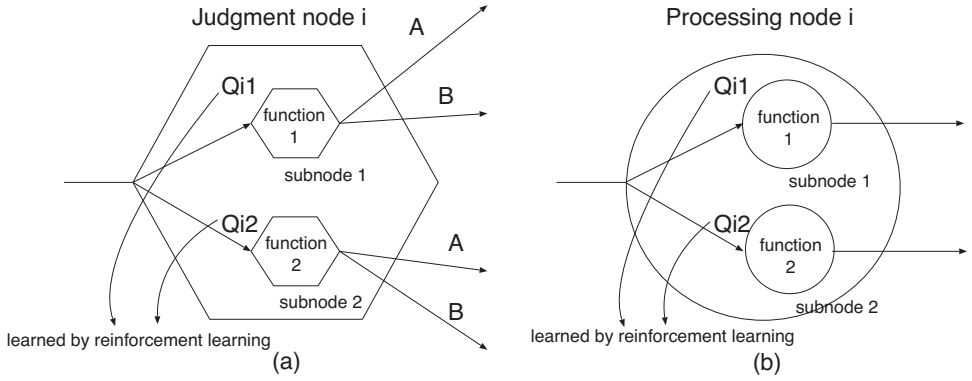



**Fig. 2.** (a) Judgment node and (b) processing node structures.

learning selects one subnode based on $\varepsilon$-greedy policy. GNP-RL effectively creates programs by using reinforcement learning and evolution (explained later).

### 2.2. Node transition and reinforcement learning

In this paper, Sarsa [24] is used as the reinforcement learning algorithm of GNP-RL. The reason why we select Sarsa is that on-policy algorithm (Sarsa) is effective to optimize state transitions considering the effect of $\varepsilon$-greedy policy.

The node transition of the graph program is carried out as follows. If the current node is a judgment node, one of the subnodes in the current node is selected according to $\varepsilon$-greedy policy. After executing the function of the selected subnode, the current node is transferred to the next node according to the judgment result (A or B in Fig. 2). If the current node is a processing node, one of the subnodes is selected by the same way as judgment nodes. After executing the function of the selected subnode, the next node is determined by the connection from the subnode. This reinforcement learning framework using graph structure corresponds to the general reinforcement learning, that is, each node is defined as "state" and the selection of a subnode is defined as "action", and the state transition will be optimized. Every node transition, $Q$ value is updated by Sarsa algorithm as follows.

$$Q_{ip} \leftarrow Q_{ip} + \alpha(r + \gamma Q_{jq} - Q_{ip}), \tag{1}$$

where $Q_{ip}$ shows the $Q$ value of the selected subnode $p$ at node $i$, $Q_{jq}$ shows the $Q$ value of the selected subnode $q$ at the next node $j$, and $r$ is a reward obtained after executing node $i$. $\alpha(\in(0, 1.0])$ is a learning rate, $\gamma(\in[0, 1.0])$ is a discount rate.

### 2.3. Evolution

Evolution is realized by crossover and mutation. In the crossover, two individuals are selected as parents by tournament selection [25], then randomly selected nodes with the probability of $P_c$ are exchanged between two parents. In the mutation, one individual is selected by tournament selection, then randomly selected node branches with the probability of $P_m$ are re-connected to other nodes. In addition, randomly selected node functions with the probability of $P_m$ are also changed to other functions. More detailed explanation is described in [11].

### 2.4. Rule pool generation

As described in Section 1, the different concept from general evolutionary algorithms is adopted in the rule pool generation. General evolutionary algorithms aim to evolve individuals and the elite individual obtained in the final generation becomes the solution of the problem. However, in the rule pool generation of GNP-RL, a large number of rules are extracted by the elite individuals throughout the generations.

Here, a rule used in this paper is defined. Let $\{X_1, X_2, X_3, \dots\}$ be input attributes, and $\{Y\}$ be a decision class attribute. Then, for example, the following rule can be generated.

*If* $X_1 = A$ and $X_2 = B$ and $X_3 = C$, Then $Y = D$,

where, $A$, $B$, $C$ and $D$ show the attribute values. If the antecedent part of this rule is satisfied, that is, the current situation is "$X_1$ is $A$, $X_2$ is $B$, and $X_3$ is $C$", then the decision is made by the consequent part of the rule, that is, "$Y$ must-be $D$".

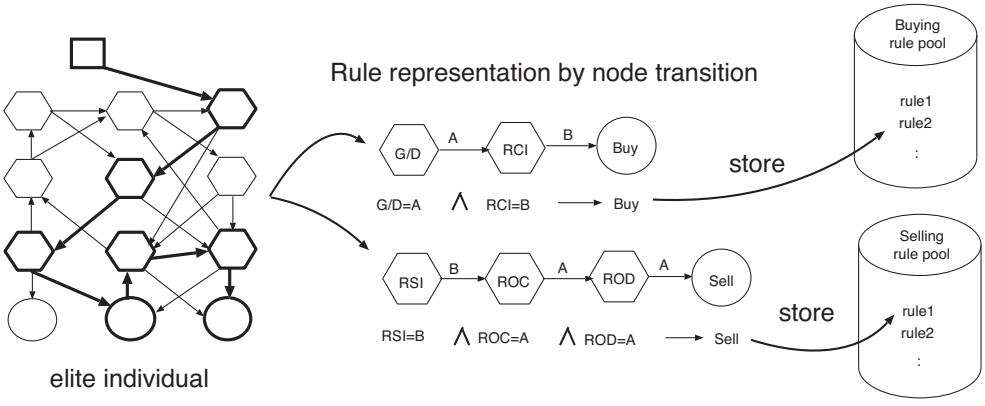



**Fig. 3.** Rule representation.

**Table 1**
Calculation periods of the technical indexes [day].

| Technical index | Calculation period | | |
|---|---|---|---|
| | Period 1 | Period 2 | Period 3 |
| Rate of deviation (ROD) | 5 | 10 | 25 |
| Relative strength index (RSI) | 5 | 10 | 25 |
| Rate of change (ROC) | 5 | 10 | 25 |
| Volume ratio | 5 | 10 | 25 |
| Stochastics | 5 | 10 | 25 |
| Rank correlation index (RCI) | 5 | 10 | 25 |

| Technical index | Calculation period |
|---|---|
| Golden/dead cross | 5 (short term), 25 (long term) |
| MACD (moving average convergence divergence) | 5 (short term), 25 (long term), 9 (signal) |
| Candle chart | – |

Every generation, the node transition of the elite individual is saved, and rules are extracted as shown in Fig. 3, where G/D (golden/dead cross), RCI (rank correlation index), RSI (relative strength index), ROC (rate of change) and ROD (rate of deviation) are the technical indexes (input attributes) shown in Table 1.

A rule consists of successive judgments with their judgment results (antecedent part) and the resulting processing (consequent part). In the stock trading problem dealt with in this paper, the antecedent part consists of technical indexes of stocks, and the consequent part shows buying or selling action. In addition, rule pools are prepared for each consequent, that is, buying rule pool and selling rule pool are prepared. If the same rules which have been already stored in the rule pools are extracted by the elite individuals in other generations, such rules are not stored in the rule pools again.

In order to clearly show the rule forming mechanism, the relation between a program (graph structure) of GNP-RL and the rule extraction is explained using an example shown in Fig. 4. The table in Fig. 4 shows an example of data at time $t$, $t+1$ and $t+2$, and the three graph structures in Fig. 4 show the node transitions executed at time $t$, $t+1$ and $t+2$, respectively. At time $t$, the node transition starts from the start node and the current node transfers to the next node $X_1$. The value of $X_1$ is $A$ according to the table, thus the branch $A$ is selected and the current node transfers to the next node $X_2$. As the same way as the previous node, the attribute value of $X_2$ is checked in the table ($X_2 = A$), then the branch $A$ is selected to transfer to the next node $X_3$. After checking the value of $X_3$, the current node transfers to the node "$Y = buy$", thus the action "buy" is decided at time $t$. As a result, a rule "If $X_1 = A$ and $X_2 = A$ and $X_3 = A$, Then $Y = buy$" is generated. At time $t+1$, the node transition continues from the last node used at time $t$ and a new rule is generated. The node transitions at time $t+1$ and $t+2$ and their corresponding rules are also shown in Fig. 4. After finishing a task, all the generated rules are stored in the buying rule pool and selling rule pool. However, as described before, the rule extraction is executed by the

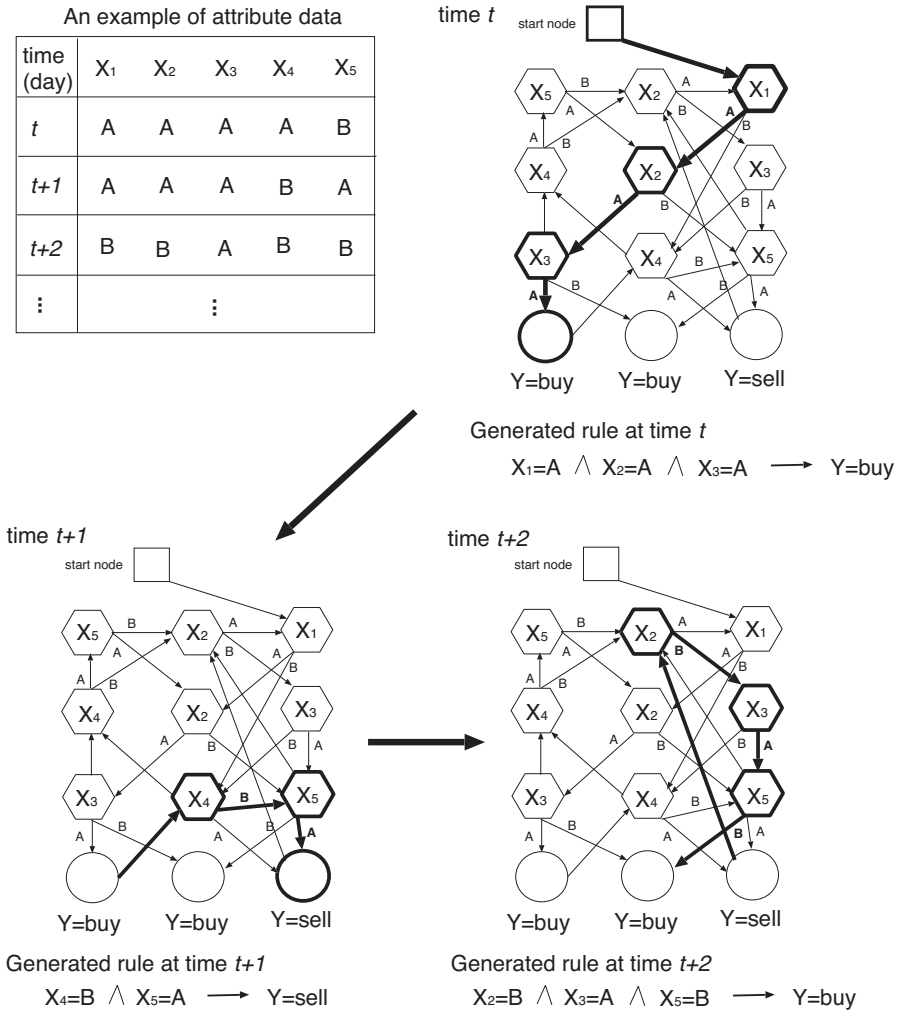



**Fig. 4.** An example of rule extraction from graph structure.

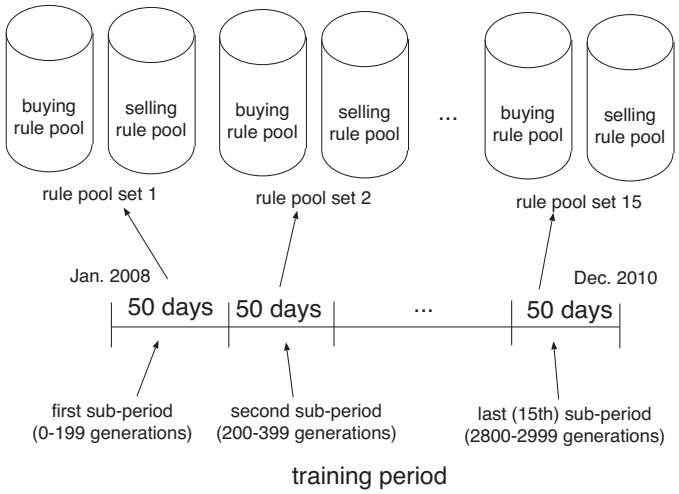



**Fig. 5.** An example of training period division.

elite individual (obtaining the highest profit in the stock trading) in each generation.

## 3. Proposed method: ensemble learning using MLP for selecting rule pools

This section explains how to make several rule pools for adapting to many kinds of stock market situations, and how to realize ensemble learning using MLP. The points of this paper are to create several rule pools (classifiers) where a large number of rules for adapting to various kinds of situations are stored, and the best sets of the rules are selected by MLP depending on the stock market conditions. As a result, the combination of the rule-based decision making model and ensemble learning using MLP can generate trading signals effectively.

### 3.1. Rule pool generation for various situations

Suppose that stock price data of three years are used as training data, and the whole period of the training data is divided into several sub-periods. For example, when the training data is divided every 50 days, the stock data in 2008–2010 in Tokyo stock exchange in Japan can be divided into 15 sub-periods (Fig. 5). Then, the rule extraction is carried out for each sub-period separately. First, graph programs evolve for 50 generations using the data of the first sub-period and the rules are stored in the rule pools for the first sub-period. After finishing 50 generations, the training data is changed from the first to second sub-period, and the program evolution continues and the extracted rules are stored in the rule pools for the second sub-period. The same procedure is repeated until the evolution and rule extraction for the last sub-period finish. As shown in Fig. 5, one set of rule pools is generated for each sub-period and the number of rule pool sets is the same as the number of sub-periods.

### 3.2. MLP structure and learning

First, the concept of the proposed ensemble learning is reviewed. In fact, there are various trends in the stock markets, thus it is very difficult to deal with all the trends using only one set of rules. Therefore, in this paper, the whole training period (data) is divided into 15 sub-periods and 15 sets of rules which well deal with each sub-period are created, which means that each set of rules is an expert rule set on the corresponding sub-period, i.e., trend. Then, the role of MLP is explained. Fig. 6 shows three layered perceptron and its relation to the rule pools. In this paper, MLP is used as a classifier, where the inputs are the values of

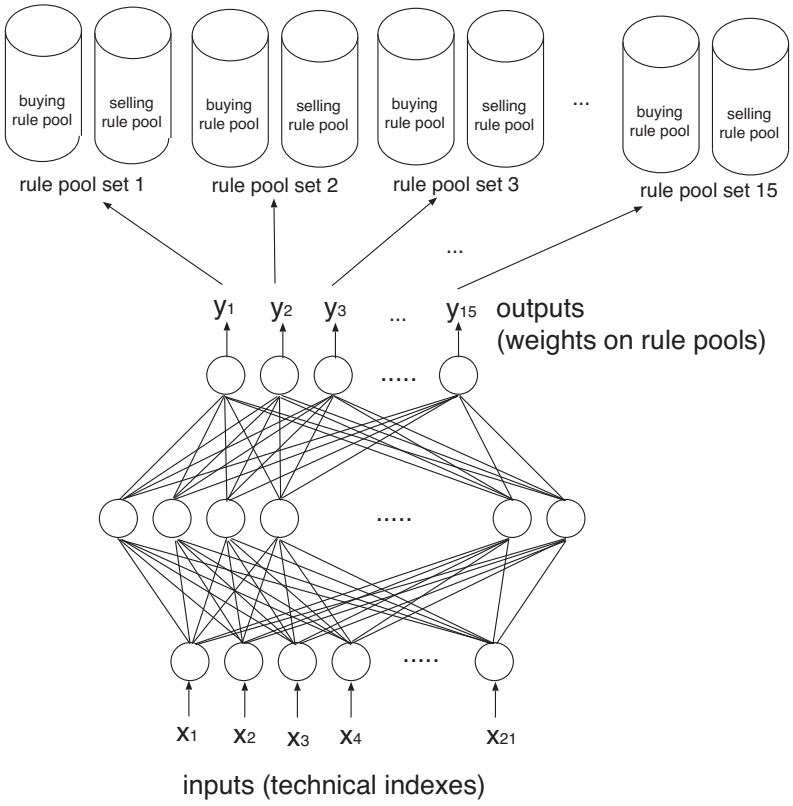
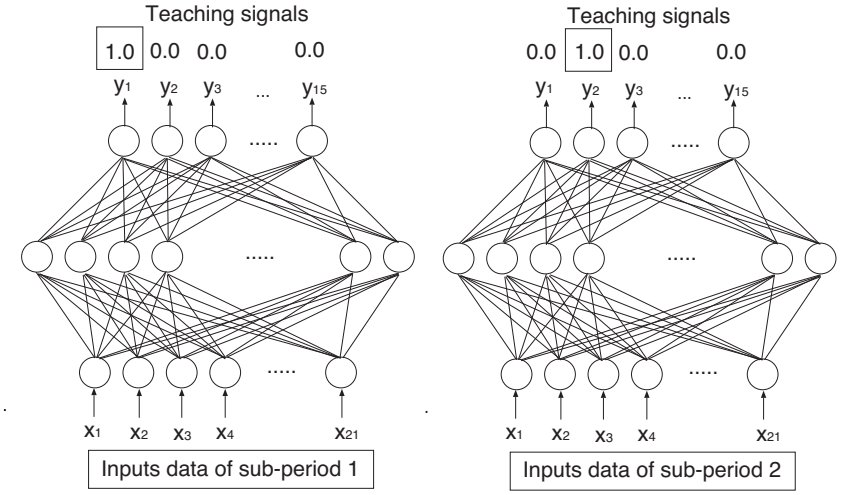
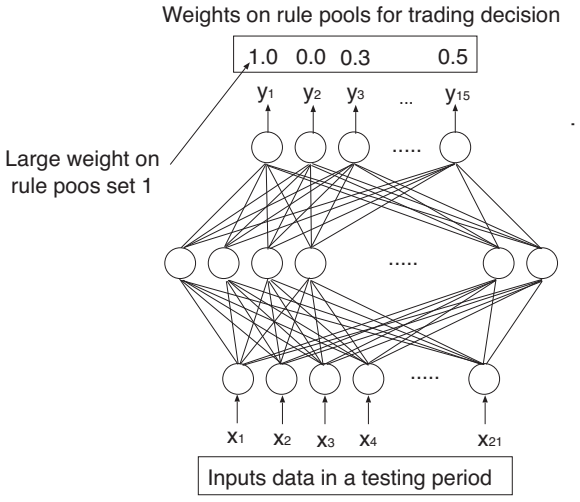



**Fig. 6.** MLP structure.




(a) Training (learning)




(b) Testing

**Fig. 7.** Examples of the MLP training and testing.

technical indexes, and the outputs are the similarity values (used as weights) to each trend (rule pool set). The learning of MLP is implemented in such a way that the output value of output node "1", i.e., $y_1$, becomes high (and other values $y_2$–$y_{15}$ become low) when the input data belong to sub-period "1" (Fig. 7 (a) left-side), the output of node "2", i.e., $y_2$, becomes high (and other values $y_1, y_3$–$y_{15}$ become low) when the input data belong to sub-period "2" (Fig. 7(a) right-side), and so on. As a result, the output values indicate the similarity degrees to each trend, thus in the testing, we can use them as weights on the sets of rules, where the expert rule sets on the current trend have large weights on the decision making (Fig. 7(b)). The uncertainty of the decision making process can be decreased by avoiding using rules that are not matched with the current stock market situation. In addition, the reason why we selected MLP for the proposed ensemble learning method is that MLP has high generalization ability for unknown data patterns. In fact, even if the MLP is trained using enough number of training data, there are possibilities that unknown input patters are given in the testing phase. In this case, the generalization ability of MLP works well for generating effective outputs.

The technical indexes used as inputs in this paper are shown in Table 1. In Table 1, eight kinds of technical indexes are shown, however, six indexes (ROD, RSI, ROC, volume ratio, Stochastics and RCI) among the eight indexes are calculated using three kinds of calculation periods, thus $18(=3\times6)$ index values are generated. Other three kinds of indexes, i.e., Golden/Dead cross, MACD and candle chart, generate just one value, respectively. Therefore, the total number of index values becomes $21(=18+3)$. The number of outputs of MLP is 15 that is the same as the number of rule pool sets, and the learning of MLP is carried out by back propagation algorithm [26]. The teaching signals for the learning of MLP are given as follows. For example, if one training data belongs to sub-period "2", the teaching signals for 15 outputs are $\{t_1, t_2, t_3, t_4, \ldots, t_{15}\} = \{0, 1, 0, 0, \ldots, 0\}$. As a result, when the testing data is fed into the learned MLP, MLP generates outputs $y_p \in [0.0, 1.0]\,(1 \le p \le 15)$ that show the weights on each rule pool set. Error function for the learning is as follows.

$$E = \frac{1}{2}\sum_{p=1}^{P}(y_p - t_p)^2, \tag{2}$$

where $p$ is the $p$-th output, and $P$ is the total number of outputs. Activation function of hidden and output units is a standard sigmoid function [27].

### 3.3. Trading decisions by weighted majority

#### 3.3.1. Definition of matching

First, the matching of data $d$ with rule $r$ is defined as follows.

$$Match(d, r) = \begin{cases} 1 & \textit{if} \text{ matching} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

If data $d$ (technical indexes on a certain day) is completely matched with the antecedent part of rule $r$, $Match(d, r)$ becomes 1.

Second, the average matching of data $d$ with all the buying rules in rule pool set $p$ ($p$-th sub-period) and that with all the selling rules in rule pool set $p$ are calculated as follows.

$$m_{p,c}(d) = \frac{1}{|R_{p,c}|} \sum_{r \in R_{p,c}} Match(d, r), \quad (4)$$

where $c \in \{buy, sell\}$ shows the rule pool types: buying or selling, and $R_{p,c}$ is a set of rules of type $c$ in rule pool set $p$.

Next, a set of rule pools $p$ that show "$m_{p,buy}(d) \geq m_{p,sell}(d)$" is defined as $P_{buy}$, and a set of rule pools that show "$m_{p,sell}(d) > m_{p,buy}(d)$" is defined as $P_{sell}$. Then, the scores of buying and selling for data $d$ are calculated by Eq. (5).

$$Score_c(d) = \sum_{p \in P_c} y_p(d), \; c \in \{buy, sell\} \quad (5)$$

where $y_p$ is the $p$-th output of MLP when data $d$ is fed into MLP as inputs.

Finally, a buying and selling decision is made as follows.

Procedure of buying and Selling decision

```
If no stocks are held
    If score_buy(d) > score_sell(d)
        buy stocks using all funds
    else
        do nothing
    end if
Else
    If score_sell(d) > score_buy(d)
        sell all the stocks
    else
        do nothing
    end if
 end if
```

## 4. Simulations

### 4.1. Simulation environments

The simulation environments of the stock trading are described in this section. In the simulations, 16 companies are selected from the first section of Tokyo exchange in Japan. The selected companies are Japanese leading companies with large market capitalization and high liquidity. The simulations are divided into training phase and testing phase, and the following two simulation periods are considered.

Simulation period 1:

- Training: Jan. 4, 2007–Dec. 30, 2008
- Testing: Jan. 4, 2010–Dec. 30, 2010

**Table 2**
Transaction cost.

| Buying/selling price [yen] | Transaction cost [yen] |
|---|---|
| 1–100,000 | 150 |
| 100,001–200,000 | 199 |
| 200,001–500,000 | 293 |
| 500,001–1,000,000 | 525 |
| 1,000,001–1,500,000 | 628 |
| 1,500,001–30,000,000 | 994 |
| 30,000,000– | 1050 |

Simulation period 2:

- Training: Jan. 4, 2008–Dec. 30, 2010
- Testing: Jan. 4, 2011–Dec. 30, 2011

We suppose that the initial funds are five million Japanese Yen. The transaction costs are considered as shown in Table 2 which refers to (C)SBI Securities Co., Ltd. (https://www.sbisec.co.jp/), one of the Japanese Internet securities companies. In the real stock trading, transaction cost has to be paid every buying and selling time. Therefore, it is important for the performance evaluation of any stock trading models to consider transaction cost and execute realistic simulations. The transaction lots in the real market are also considered, for example, one unit of stocks of Toyota motor is 100.

### 4.2. Simulation conditions

The parameters used in the learning and evolution of GNP-RL and MLP are shown in Table 3. These parameters are determined experimentally considering the fitness improvement and computational time in the training phase. The total number of individuals is 301 containing one elite individual, 179 individuals generated by mutation and 120 individuals generated by crossover. The number of nodes is 61 containing one start node, 40 judgment nodes and 20 processing nodes. When initializing the population, the node functions are randomly assigned and the connections are also randomly determined. $Q$ values are initialized at zero. The number of inputs of MLP is 21 that corresponds to the number of technical index values calculated by stock prices, and the number of outputs is 15 that corresponds to the number of rule pool sets. The number of learning iteration is 10,000 where one iteration means one set of learning for all the training data.

Eqs. (6) and (7) show the reward function used in the reinforcement learning, and the fitness function used in the evolution of GNP-RL, respectively.

$$Reward = \text{Selling price} - \text{buying price}, \quad (6)$$

**Table 3**
Simulation conditions.

| GNP-RL | |
|---|---|
| The number of individuals | 301 |
| The number of nodes | 61 |
| The number of generations | 3000 |
| Crossover rate $P_c$ | 0.1 |
| Mutation rate $P_m$ | 0.02 |
| $\alpha$ | 0.1 |
| $\gamma$ | 0.4 |
| $\varepsilon$ | 0.1 |
| **MLP** | |
| The number of inputs | 21 |
| The number of outputs | 15 |
| The number of hidden units | 50 |
| Learning rate | 0.1 |
| Iteration | 10,000 |

which shows the profit of one set of trades, i.e., one time buying and one time selling.

Fitness function is,

$$\text{Fitness} = \textit{Total profit in the training period.} \quad (7)$$

Table 4 shows the technical indexes used in the simulations and the judgment results obtained by their values. For example, if the value of rate of deviation is 0.01, the judgment result becomes "C", which means that the judgment node of rate of deviation selects branch "C". The judgment result of candle chart is determined by the eight candle patterns shown in Fig. 8 [19].

### 4.3. Simulation results

The stock trading simulations are carried out for each stock independently, and the simulation for each stock is repeated 30 times. Therefore, the values shown in the simulation results are the average over 30 independent simulations.

#### 4.3.1. Training results

Fig. 9 shows the accumulated number of rules through the generations. As the generation goes on, the number of rules increases and finally the number of buying rules becomes 2829 and that of selling rules becomes 2771. Fig. 10 shows the changes of the mean error in the MLP training using the data of Mitsui & Co. in 2008–2010. The mean error at each iteration is obtained by calculating the mean value of $E$ (Eq. (2)) for all the stock data in 2008–2010 (733 data). The testing simulations in the next section are carried out by using the extracted rules and the learned MLP.

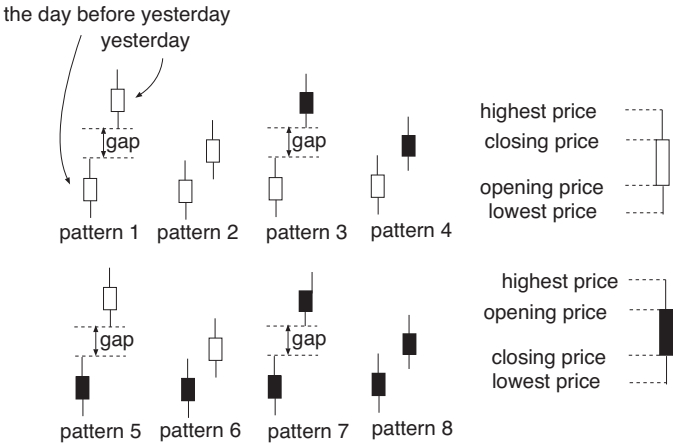



**Fig. 8.** Candle patterns.

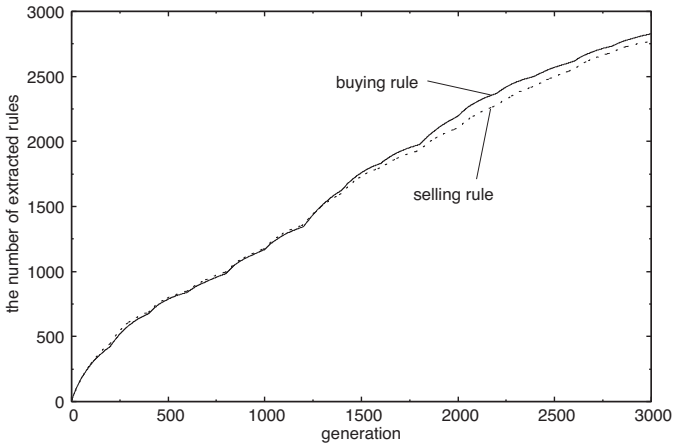



**Fig. 9.** The number of extracted rules (Toyota motor in 2008–2010).

#### 4.3.2. Testing results

First, the testing results obtained in simulation period 1 (testing year: 2010) are explained. In Table 5, the profits [Japanese yen] (and profit rates [%]) obtained by the three methods (ensemble GNP-RL, GNP-RL without ensemble learning and buy&hold) for 16 stocks (cases) are shown. Buy&hold buys stocks as much as possible at the beginning of the testing year and sells all the stocks at the end of the year. Buy&hold is considered as a benchmark of the market which shows the changes of the stock prices in the target year. Each profit obtained by ensemble GNP-RL or GNP-RL without ensemble learning in Table 5 is the average over 30 independent simulations. The profits obtained by buy&hold are calculated by the stock prices at the beginning and end of the testing period (therefore, the profits are not the average values because they are calculated by one fixed data, respectively).

When the average values are compared between the methods, statistical analysis is very important to show the significant difference in the average values, thus $t$-test between ensemble GNP-RL and GNP-RL without ensemble learning is carried out, and the $p$-values are also shown in Table 5. If the $p$-value is less than the threshold (here, 5[%]), we can say that there is a significant difference in the average values between the two methods. In the row of "Toyota Motor" in Table 5, for example, the proposed method obtains the profit of −427,644 [yen] (profit rate is −8.6[%]); GNP-RL without ensemble learning obtains the profit of -23,695 [yen] (profit rate is -0.5); and the p-value is $1.4 \times 10^{-1}$[%] (<5 [%]); thus in this case, GNP-RL without ensemble learning is better. However,

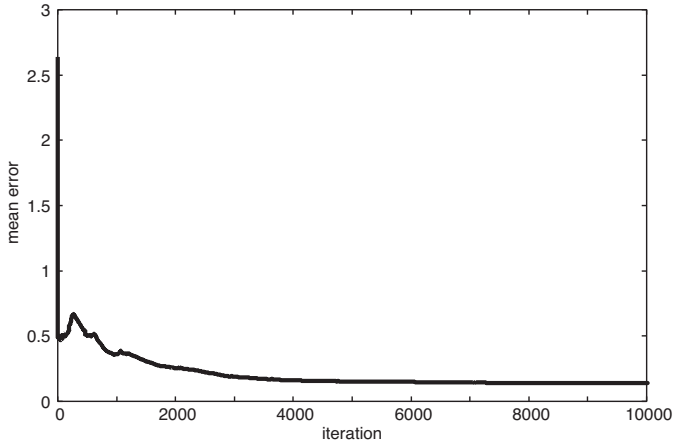



**Fig. 10.** Mean error in MLP training (Mitsui & Co. in 2008–2010).

**Table 4**
Judgment results for each technical index and candle patterns.

| Technical index | Judgment result | | | | |
|---|---|---|---|---|---|
| | A | B | C | D | E |
| 1. Rate of deviation | $[-1, -0.1]$ | $(-0.1, -0.05]$ | $(-0.05, 0.05)$ | $[0.05, 0.1)$ | $[0.1, \infty)$ |
| 2. RSI | $[0, 0.2]$ | $(0.2, 0.8)$ | $[0.8, 1]$ | – | – |
| 3. ROC | $[0, 0.9)$ | $[0.9, 1.1]$ | $(1.1, \infty)$ | – | – |
| 4. Volume ratio | $[0, 0.3)$ | $[0.3, 0.7]$ | $(0.7, 1]$ | – | – |
| 5. Stochastics | $[0, 0.3]$ | $(0.3, 0.7)$ | $[0.7, 1]$ | – | – |
| 6. RCI | $[-1, -0.7]$ | $(-0.7, 0.7)$ | $[0.7, 1]$ | – | – |
| 7. Golden/dead cross | Golden cross | Dead cross | The other | – | – |
| 8. MACD | Golden cross | Dead cross | The other | – | – |
| 9. Candle chart | Each pattern in Fig. 8 corresponds to a judgment result | | | | |

**Table 5**
Profits in the testing (2010).

| Stock | Ensemble GNP-RL profit [yen] (profit rate [%]) | | GNP-RL w/o ensemble profit [yen] (profit rate [%]) | | Buy&hold profit [yen] (profit rate [%]) | | $p$-Value [%]† |
|---|---|---|---|---|---|---|---|
| Toyota Motor | −427,644 | (−8.6) | −23,695 | (−0.5) | −793,988 | (−15.9) | $1.4 \times 10^{-1}$ |
| Mitsubishi Estate | 25,773 | (0.5) | −707,231 | (−14.1) | 142,012 | (2.8) | $2.9 \times 10^{-8}$ |
| Showa Shell | 216,260 | (4.3) | 581,832 | (11.6) | −138,488 | (−2.8) | $3.4 \times 10^{-2}$ |
| East Japan Railway | −348,048 | (−7.0) | −156,987 | (−3.1) | −425,988 | (−8.5) | $2.3 \times 10^{-1}$ |
| NEC | 539,855 | (10.8) | 129,563 | (2.6) | 166,012 | (3.3) | $3.3 \times 10^{-1}$ |
| Fuji Heavy Ind. | 1,435,308 | (28.7) | 810,160 | (16.2) | 2,033,012 | (40.7) | $5.6 \times 10^{-3}$ |
| Sekisui House | 519,351 | (10.4) | −213,359 | (−4.3) | −175,988 | (−3.5) | $4.8 \times 10^{-6}$ |
| Mitsui &Co. | 1,082,589 | (21.7) | −518,425 | (−10.4) | 58,012 | (1.2) | $1.1 \times 10^{-14}$ |
| Sony | 572,279 | (11.4) | −154,120 | (−3.1) | 433,612 | (8.7) | $5.1 \times 10^{-4}$ |
| Tokyo Gas | −140,057 | (−2.8) | −165,605 | (−3.3) | −92,988 | (−1.9) | 37 |
| KDDI | 335,044 | (6.7) | −180,273 | (−3.6) | −195,988 | (−3.9) | $4.3 \times 10^{-3}$ |
| Tokyo Electric Power | −516,877 | (−10.3) | −723,173 | (−14.5) | −724,388 | (−14.5) | $7.9 \times 10^{-2}$ |
| Daiwa House | −169,019 | (−3.4) | 26,491 | (0.5) | 10,012 | (0.2) | 2.8 |
| Nomura Holdings | −748,102 | (−15.0) | −1,009,247 | (−20.2) | −1,161,188 | (−23.2) | 2.2 |
| Shin-Etsu Chemical Co., Ltd. | −503,804 | (−10.1) | −801,404 | (−16.0) | −780,488 | (−15.6) | $1.2 \times 10^{-1}$ |
| Nippon Steel Cooperation | −425,055 | (−8.5) | −447,701 | (−9.0) | −1,067,988 | (−21.4) | 42 |
| Mean | 90,491 | (1.8) | −222,073 | (−4.4) | −169,676 | (−3.4) | |

† $p$-value shows the result of $t$-test between ensemble GNP-RL and GNP-RL w/o ensemble

in the row of "Mitsubishi Estate", the proposed method obtains the profit of 25,773 [yen] (profit rate is 0.5 [%]); GNP-RL without ensemble learning obtains the profit of −707,231 [yen] (profit rate is −14.1 [%]); and the $p$-value is $2.9 \times 10^{-8}$ [%](<5[%]); thus in this case, the proposed method is better with significant difference. If the $p$-value is 5 [%] or larger, it means that there is no significant difference between the two methods. In this way, the profits obtained in the 16 cases can be analyzed, and we can find that the proposed method shows better profits than GNP-RL without ensemble learning in 10 cases out of 16, and almost the same results in two cases. The profits of buy&hold are not the average value, thus we simply compare the profits in Table 5 between the proposed method and buy&hold. Then, the proposed method shows better profits than buy&hold in 12 cases out of 16. Considering the above results, it is clarified that the ensemble learning using MLP enhances the performance of GNP-RL. Especially, although the results of buy&hold in

**Table 6**
Profits in the testing (2011).

| Stock | Ensemble GNP-RL profit [yen] (profit rate [%]) | | GNP-RL w/o ensemble profit [yen] (profit rate [%]) | | Buy&hold profit [yen] (profit rate [%]) | | $p$-Value [%]† |
|---|---|---|---|---|---|---|---|
| Toyota Motor | −595,721 | (−11.9) | −933,574 | (−18.7) | −1,072,988 | (−21.5) | $5.8 \times 10^{-1}$ |
| Mitsubishi Estate | −1,010,631 | (−20.2) | −215,849 | (−4.3) | −1,174,988 | (−23.5) | $3.5 \times 10^{-4}$ |
| Showa Shell | −1,593,094 | (−31.9) | −682,289 | (−13.6) | −1,526,588 | (−30.5) | $1.4 \times 10^{-3}$ |
| East Japan Railway | 175,603 | (3.5) | −485,565 | (−9.7) | −429,488 | (−8.6) | $2.7 \times 10^{-3}$ |
| NEC | −1,381,768 | (−27.6) | −1,120,217 | (−22.4) | −1,841,988 | (−36.8) | 1.7 |
| Fuji Heavy Ind. | −691,788 | (−13.8) | −1,503,149 | (−30.1) | −1,219,988 | (−24.4) | $4.2 \times 10^{-8}$ |
| Sekisui House | −889 | (−0.0) | −572,322 | (−11.4) | −943,988 | (−18.9) | $4.2 \times 10^{-4}$ |
| Mitsui &Co. | 429,705 | (8.6) | −385,435 | (−7.7) | −505,988 | (−10.1) | $1.7 \times 10^{-6}$ |
| Sony | −799,497 | (−16.0) | −1,676,050 | (−33.5) | −2,544,388 | (−50.9) | $7.4 \times 10^{-4}$ |
| Tokyo Gas | −58,697 | (−1.2) | −225,425 | (−4.5) | −144,988 | (−2.9) | 5.0 |
| KDDI | −265,878 | (−5.3) | 977,686 | (19.6) | 233,212 | (4.7) | $2.9 \times 10^{-8}$ |
| Tokyo Electric Power | −4,273,298 | (−85.5) | −3,758,949 | (−75.2) | −4,533,787 | (−90.7) | $1.0 \times 10^{-1}$ |
| Daiwa House | −573,257 | (−11.5) | −154,347 | (−3.1) | −353,988 | (−7.1) | $3.2 \times 10^{-4}$ |
| Nomura Holdings | −1,284,538 | (−25.7) | −2,191,793 | (−43.8) | −2,766,488 | (−55.3) | $2.8 \times 10^{-3}$ |
| Shin-Etsu Chemical Co., Ltd. | −272,237 | (−5.4) | −240,480 | (−4.8) | −766,488 | (−15.3) | 39 |
| Nippon Steel Cooperation | −864,091 | (−17.3) | −1,160,781 | (−23.2) | −1,665,988 | (−33.3) | 1.4 |
| Mean | −816,255 | (−16.3) | −895,534 | (−17.9) | −1,328,682 | (−26.6) | |

† $p$-value shows the result of $t$-test between ensemble GNP-RL and GNP-RL w/o ensemble

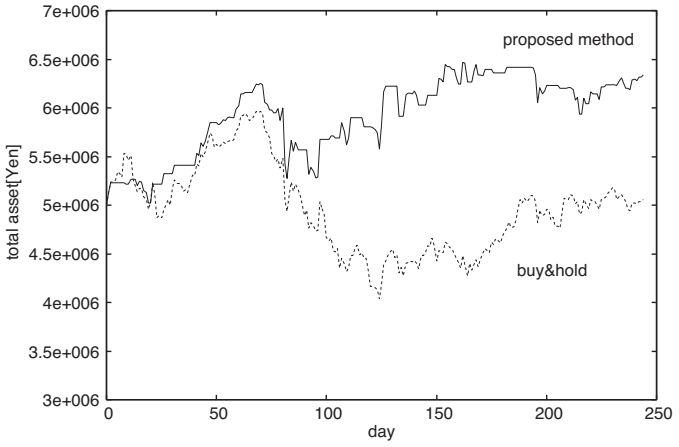



**Fig. 11.** An example of the changes of total asset in the testing phase (Mitsui & Co. in 2010).

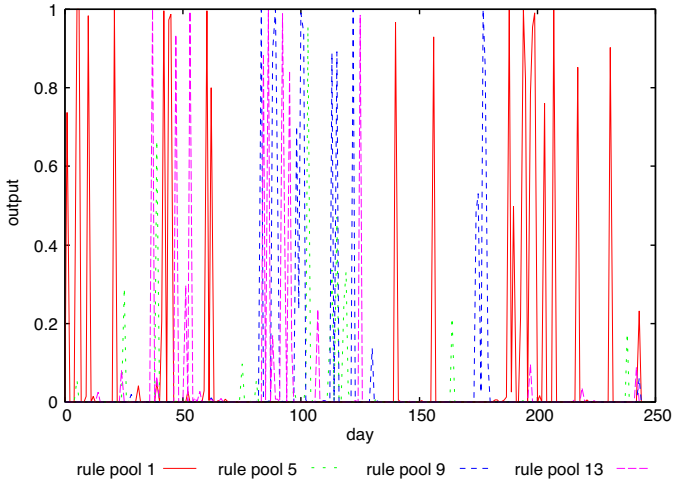



**Fig. 12.** Changes of MLP outputs in the testing phase (Mitsui & Co. in 2010).

some cases (Showa shell, Sekisui house, KDDI) show negative profits, that is, the market is down trend, the proposed method shows the positive profits.

Next, the testing results obtained in simulation period 2 (testing year: 2011) are explained. Table 6 shows the profits [Japanese yen] (and profit rates [%]) obtained by the three methods and the results of t-test between the proposed method and GNP-RL without ensemble learning. In this year, the big earthquake in the east Japan occurred, therefore, the stock market showed high volatility and the strong downtrend was induced. In Table 6, the proposed method shows higher profits or lower losses than GNP-RL without ensemble learning in nine cases with significant differences, and almost the same results in one case. The proposed method shows higher profits or lower losses than buy&hold in 13 cases. As the results of buy&hold show, the big downtrend causes large losses, however, the proposed method can reduce the losses and make profits in some cases, therefore, the total losses become lower than the other methods. Nevertheless, it is still very difficult situation for the stock trading in 2011 because the earthquake is an unpredictable phenomenon and quick response to the abnormal situation is still very difficult. Therefore, to make trading rules considering various kinds of risks would be one of the future research for creating secure models.

Fig. 11 shows an example of the changes of total asset obtained by the proposed method and buy&hold in the testing phase, where the line of buy&hold can be regarded as the changes of stock prices, i.e., market trend. It can be seen from Fig. 11 that the proposed method shows better result at the end of the testing phase, and especially, it can avoid the losses in the down trend. For example, around day 80–120, the market price is getting lower, but the proposed method keeps profit by not holding stocks during the down trend and earns profits in the local up trends. The proposed method can appropriately judge the trends of markets by giving large weights on the rule pools that match with the current situation.

Finally, the output values generated by MLP in the testing phase are analyzed using Fig. 12. Although MLP has 15 outputs, Fig. 12 concentrates on four output values for rule pool 1, 5, 9 and 13 for the analysis because too many lines overlapped with each other lose the readability of the figure. The stock (Mitsui & Co.) and the trading period (2010) dealt with in Fig. 12 are the same as Fig. 11, therefore, the outputs of MLP in Fig. 12 have a relation to the result of the proposed method in Fig. 11. It can be seen from Fig. 12 that the output values for rule pool 1 are high at the beginning (day 0–60) and the end (day 180–240) of the testing phase, and the output values for rule pool 9 and 13 are high especially around day 80–130. The periods of day 0–60 and 180–240 are basically up trends as shown by the result of buy&hold in Fig. 11, and the period of day 80–130 is basically big down trend, therefore, the interesting point of these results is that the proposed method determines the effective rule pools depending on the trends of the market.

## 5. Conclusions

This paper proposes an ensemble learning mechanism of rule-based evolutionary algorithm using MLP to effectively make decisions combining plural number of rule pools (classifiers). MLP can find the similar situations in the training phase to the situations in the testing phase, and similarity values (weights) can be given to the rule pools. Then, the final decision is executed by the weighted scores of buying and selling actions, thus the trading decision can be made by important rules that are suitable for the current situation. From the simulation results, the proposed method with ensemble learning shows better results than that without ensemble learning and buy&hold, therefore, the effectiveness of the ensemble learning using MLP for stock trading problems is clarified. In the future, distributed and hierarchical structures of evolutionary algorithms will be studied to extract more effective rules for adapting to various kinds of situations, and sophisticated data mining algorithms will be also applied to make more secure models. In addition, we will consider the comparisons with other techniques for the ensemble learning by extending this research.

## References

[1] N. Baba, N. Inoue, Y. Yanjun, Utilization of soft computing techniques for constructing reliable decision support systems for dealing stocks, in: Proc. of the International Joint Conference on Neural Networks, 2002, pp. 2150–2155.
[2] K.J. Oh, T.Y. Kim, S.-H. Min, H.Y. Lee, Portfolio algorithm based on portfolio beta using genetic algorithm, Expert Syst. Appl. 30 (2006) 527–534.
[3] J.-Y. Potvin, P. Soriano, M. Vallee, Generating trading rules on the stock markets with genetic programming, Comput. Oper. Res. 31 (2004) 1033–1047.
[4] W. Huang, Y. Nakamori, S.Y. Wang, Forecasting stock market movement direction with support vector machine, Comput. Oper. Res. 32 (10) (2005) 2513–2522.
[5] R. Li, T.-P. Tian, S. Sclaroff, Divide, conquer and coordinate: globally coordinated switching linear dynamical system, IEEE Trans. Pattern Anal. Mach. Intell. 34 (2012) 654–669.
[6] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[7] B. Efron, R.J. Tibshirani, An Introduction to the Bootstrap, Springer, 1993.
[8] Y. Freund, R.E. Schapire, A desicion-theoretic generalization of on-line learning and an application to boosting, in: Computational Learning Theory, Springer, 1995, pp. 23–37.
[9] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.

[10] K. Hirasawa, M. Okubo, H. Katagiri, J. Hu, J. Murata, Comparison between genetic network programming (GNP) and genetic programming (GP), in: Proc. of the Congress on Evolutionary Computation, 2001, pp. 1276–1282.
[11] S. Mabu, K. Hirasawa, J. Hu, A graph-based evolutionary algorithm: Genetic network programming (GNP) and its extension using reinforcement learning, Evol. Comput. 15 (3) (2007) 369–398.
[12] J.R. Koza, Genetic Programming, on the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, MA, 1992.
[13] A.S. Ullah, K.H. Harib, A human-assisted knowledge extraction method for machining operations, Adv. Eng. Inform. 20 (4) (2006) 335–350, http://dx.doi.org/10.1016/j.aei.2006.07.004 http://www.sciencedirect.com/science/article/pii/S1474034606000486
[14] A.S. Ullah, K.H. Harib, Knowledge extraction from time series and its application to surface roughness simulation, Inf. Knowl. Syst. Manage. 5 (2) (2006) 117–134.
[15] K. Hirasawa, T. Eguchi, J. Zhou, L. Yu, S. Markon, A double-deck elevator group supervisory control system using genetic network programming, IEEE Trans. Syst. Man Cybern. C 38 (4) (2008) 535–550.
[16] K. Shimada, K. Hirasawa, J. Hu, Genetic network programming with acquisition mechanisms of association rules, J. Adv. Comput. Intell. Intell. Inform. 10 (1) (2006) 102–111.
[17] K. Shimada, K. Hirasawa, J. Hu, Class association rule mining with chi-squared test using genetic network programming, in: Proc. of the IEEE International Conference on Systems, Man and Cybernetics, 2006, pp. 5338–5344.
[18] S. Mabu, C. Chen, N. Lu, K. Shimada, K. Hirasawa, An intrusion-detection model based on fuzzy class-association-rule mining using genetic network programming, IEEE Trans. Syst. Man Cybern. C 41 (1) (2011) 130–139.
[19] S. Mabu, K. Hirasawa, M. Obayashi, T. Kuremoto, Enhanced decision making mechanism of rule-based genetic network programming for creating stock trading signals, Expert Syst. Appl. 40 (2013) 6311–6320.
[20] L. Wang, S. Mabu, F. Ye, S. Eto, X. Fan, K. Hirasawa, Genetic network programming with rule accumulation and its application to tile-world problem, J. Adv. Comput. Intell. Intell. Inform. 13 (5) (2009) 551–560.
[21] A.C. Briza, P.C. Naval Jr., Stock trading system based on the multi-objective particle swarm optimization of technical indicators on end-of-day market data, Appl. Soft Comput. 11 (1) (2011) 1191–1201.
[22] P.-C. Chang, T.W. Liao, J.-J. Lin, C.-Y. Fan, A dynamic threshold decision system for stock trading signal detection, Appl. Soft Comput. 11 (5) (2011) 3998–4010.
[23] H.-H. Chen, C.-B. Yang, Y.-H. Peng, The trading on the mutual funds by gene expression programming with sortino ratio, Appl. Soft Comput. 15 (2014) 219–230.
[24] R.S. Sutton, A.G. Barto, Reinforcement Learning – An Introduction, MIT Press, Cambridge, MA/London, England, 1998.
[25] T. Bäck, D.B. Fogel, Z. Michalewicz, Evolutionary Computation 1: Basic Algorithms and Operators, CRC Press, 2000.
[26] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning Representations by Back-Propagating Errors, MIT Press, Cambridge, MA, USA, 1988.
[27] C.M. Bishop, Neural Networks for Pattern Recognition, Clarendon Press, Oxford, 1995.