# Trading Strategy Mining with Gene Expression Programming*

Chang-Hao Huang, Chang-Biau Yang† and Hung-Hsin Chen
*Department of Computer Science and Engineering*
*National Sun Yat-sen University*
*Kaohsiung 80424, Taiwan*

***Abstract*—In the paper, we study the investment on Taiwan Stock Exchange Capitalization Weighted Stock Index (TAIEX), which is assumed to be tradable. We apply the gene expression programming (GEP) to mining profitable trading strategies in the training phase. GEP is a good tool for evolving formulas since the logical view of its chromosome is a tree structure and the physical implementation is a linear string. In the testing phase, we find out some template intervals from historical data series which are similar to the leading interval. Then the trading strategies extracted from the template intervals are invoked to determine the trading signal. To keep stability of investment return, we perform a simple majority vote on a set of trading strategies. The testing period contains TAIEX starting from 2000/9/14 and ending on 2012/1/17, more than eleven years. In our experiments, the lengths of training intervals are 60, 90, 120, 180, and 270 trading days. The best cumulative return 236.25% and the best annualized return 10.63% occur when each training interval has 180 days, which are higher than the cumulative return 0.96% and annualized return 0.08% of the buy-and-hold strategy. The experimental results show that our method with higher voting threshold can usually make profitable trading decisions.**

***Keywords*-Taiwan Stock Exchange Capitalization Weighted Stock Index (TAIEX); gene expression programming; majority vote; feature set; strategy pool.**

## I. Introduction

To get high return on the investment of the stock market is a challenging task, since the stock market fluctuates frequently. If an investor buys stocks at a local valley and sells stocks at a local peak during the stock price variation, he will gain a great amount of return. However, it is very difficult for an investor to decide whether the trend of the stock price is rising or falling in the future.

Many machine learning techniques have been applied to maximization of the return on the investment of the stock market, such as the *artificial neural networks* (ANN) for predicting the close price or the trading signal in the next trading day [7], [2], the *genetic programming* (GP) [3], [8] and *genetic algorithm* (GA) [11] for evolving the trading strategy to gain return, and the *support vector machine* (SVM) [10] for predicting the trend of stock market in the future. Furthermore, some studies [1], [2], [5] used distance-oriented methods, such as *root mean square error*, *cosine distance* and *dynamic time warping*, to retrieve time series patterns from historical data and to construct trading signals by utilizing these patterns. Also, some studies [7], [10] utilized the bionics methods, such as *artificial neural network* [7] and *swarm optimization* [12], to select important features of the stock market, and then the trading system was trained with these features to predict whether the stock price in the next trading day is rising or falling.

In the paper, we confine our investment in Taiwan stock market, and our goal is to earn stable return on investment in *Taiwan Stock Exchange Capitalization Weighted Stock Index* (TAIEX). In the training phase, *gene expression programming* (GEP) is used to generate profitable trading strategies for the training period. To increase the quality of the trading decisions in the testing period, we adopt the simple majority vote method, in which the votes contain the signals generated by the pre-trained trading strategies.

The dataset for our experiments contains TAIEX starting from 1995/1/5 and ending on 2012/1/17. The training period starts on 1995/1/5 and ends on 2012/1/17, and the testing period is from 2000/9/14 to 2012/1/17. In our experiments, the lengths of training intervals are 60, 90, 120, 180, and 270 trading days. The best cumulative return 236.25% and the best annualized return 10.63% occur when the training interval is 180 days with available threshold 0.21 and voting threshold 0.88. This result is better than the cumulative return 0.96% and annualized return 0.08% of the buy-and-hold strategy. The experimental results show that the model with higher voting threshold usually has better return.

The remainder of this paper is organized as follows. In Section II, we will introduce the gene expression programming (GEP) and review the previous work proposed by Jhou *et al.*'s [8]. In Section III, we will propose our strategy mining method with GEP in the training phase. Section IV presents the method for determining the trading signal in the testing phase. In Section V, we will show the experimental results and compare them to the previous result. Finally, in Section VI, we will give the conclusion.

## II. Preliminaries

In this section, we will introduce the gene expression programming and the previous work proposed by Jhou *et al.* [8].

### A. Gene Expression Programming

*Genetic algorithm* (GA) [6], *genetic programming* (GP) [9] and *gene expression programming* (GEP) [4] are all evolutionary methods, which simulate the evolution process of nature creatures, the fittest to the environment

*This research work was partially supported by the National Science Council of Taiwan under contract NSC101-2221-E-110-020-MY3.
†Corresponding author: cbyang@cse.nsysu.edu.tw

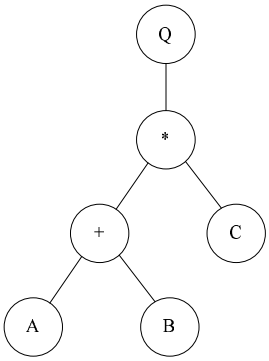


Figure 1. The expression tree of formula $\sqrt{(A+B)\times C}$, where $Q$ represents the square root function.

would survive. The chromosome structure of GA is linear, which is enough for solving many practical problems. However, some problems may not be easily encoded in a linear manner. The chromosome of GP is represented by an *expression tree* (ET), which is more flexible than GA. The tree representation of GP can be further explained as a formula. Figure 1 shows an example of ET.

Though GP has more flexibility, its weakness is the difficult implementation of the tree operations, such as cross over and mutation. The *gene expression programming* (GEP) was first proposed by Ferreira [4]. GEP is a smart implementation of GP. In GEP, a string is used as a basic data structure for representing an expression tree so that a GEP chromosome (gene) is also of linear structure. For example, the ET of Figure 1 can be translated into a coding sequence (gene) `Q*+CAB` by straightforward reading the ET from top to bottom and from left to right. On the other side, the gene can be translated back to the ET easily. If a gene contains more redundant symbols, we just remove these tail symbols. For example, `Q*+CAB*DE` has the same meaning as `Q*+CAB` since `*DE` in the former is illegal. With this flexible representation, the operations of GEP can be implemented easily and the required execution time can be reduced drastically.

One GEP chromosome may contain one or more genes of equal head length, and each gene is composed of a head and a tail. The symbols in the head are randomly selected from the function set and the terminal set, while the symbols in the tail are selected only from the terminal set. The function set usually contains operators such as arithmetic operators, Boolean operators, comparison operators, and logical operators. The terminal set usually contains random constants and independent variables for a specific problem. The evolution operations of GEP [4] include mutation, insertion sequence transposition, root insertion sequence transposition, gene transposition rate, 1-point recombination rate, 2-point recombination rate, and gene recombination.

The fitness function of GEP is utilized to measure the goodness of a chromosome in the population. One chromosome with higher score in the population has higher probability to be chosen to progress in the next step. Furthermore, to apply GEP, users have to specify the parameters and termination criteria. The parameters include the head length, population size, number of generations, rates of various evolution operations. The termination criteria usually depend on the number of generations and the convergence of the fitness score. When GEP terminates, the best evolved solution is the chromosome with the best score.

### B. Jhou's Method

Jhou *et al.*'s method [8] uses TAIEX as the investment target. Their method applies GP to evolving the best trading strategy in each testing interval. It first retrieves three historical stock price series which are the most similar to the training interval with the measure of root mean square error (RMSE). Then the technical indicators of these historical stock price series are fed into GP for generating profitable trading strategies. After finding the best strategy in the training interval, the strategy is applied to the testing interval.

Six kinds of basic information and 25 technical indicators are included as the variables of GP. To earn more stable returns, two arithmetic trees are involved in the training phase of GP. The dataset contains TAIEX from 2000/9/14 to 2010/5/21. For various combinations of training intervals and testing intervals, the experiment is performed for ten times to get the average performance. Their best cumulative return is 166.57% when the training interval has 545 days and the testing interval has 356 days, with the rolling of sliding windows.

## III. Trading Strategy Mining in the Training Period

In this paper, we propose a method that can dynamically determine the trading strategy in each trading day so that the trading decision is profitable. We first build the strategy pool for storing profitable trading strategies of all training intervals, then we search for suitable strategies for the current testing day from the pool of the historical period.

To accelerate the evolution process of trading strategies in the testing phase, we build the trading strategy pool of the training period. In the pool, the top ten best trading strategies of each training interval are preserved. Each *training interval* is a time interval of length $n$ extracted from the training period. Since the training process is very time-consuming, we extract the training intervals every ten (trading) days. In other words, if the current training interval starts on day $i$, which locates at $[i, i+n-1]$, then the next interval will start on day $i+10$, which locates at $[i+10, i+n+9]$. Before the training process, the data of the training period is normalized since it is a crucial process in our method. In each training interval, the normalized data are fed into the GEP for generating profitable trading strategies. After the training process finishes for one interval, the training interval is shifted to the next one by ten days.

To perform GEP, we have to define the function set and the terminal set in advance. In our method, the function set consists of operators '$>$', '$<$', '$=$', '$\geq$', '$\leq$', 'and', 'or',

'+', '−', '×', and '÷'. The terminal set is composed of 7 kinds of *basic information*, including the open price, close price, highest price, loweset price, trading volume, trading value and return of investment, and 25 *technical indicators*, including the MTM (momentum index), OBV (on balance volume), DI (demand index), average 5-day trading volume, average 20-day trading volume, 5-day RSI (relative strength index), 14-day RSI, 10-day MA (moving average), 14-day MA, 20-day MA, TAPI (total amount per weighted stock price index), 14-day PSY (psychological), 5-day WMS (Williams), 9-day WMS, 10-day BIAS, 14-day BIAS, 20-day BIAS, 5-day OSC (oscillator), 9-day RSV (raw stochastic value), 3-day K line, 3-day D line, 12-day EMA (exponential moving average), 26-day EMA, DIF (difference), and 9-day MACD, and a constant generated randomly between -1.0 and 1.0. In addition, to evolve profitable and stable trading strategies, the 32 features on days $t$, $(t-2)$, $(t-4)$, $(t-9)$ and $(t-19)$ are also adopted in the evolution, where $t$ denotes the current trading day. Therefore, we have totally $32\times 5+1=161$ features.

The chromosome of the GEP in our method is composed of two genes, which are expressed as the *buy-tree* and the *sell-tree*, respectively. The trading signal is generated by the two expression trees, and the trading rule of day $t$ is described as follows.

$$\begin{aligned}
&\text{IF } (\textit{the value of buy-tree of day } t>0)\\
&\text{ AND } (\textit{the value of sell-tree of day } t\le 0)\\
&\qquad \text{THEN } \textit{signal}(t)\leftarrow \textit{BUY}\\
&\text{ELSE IF } (\textit{the value of buy-tree of day } t\le 0)\\
&\text{ AND } (\textit{the value of sell-tree of day } t>0)\\
&\qquad \text{THEN } \textit{signal}(t)\leftarrow \textit{SELL}\\
&\text{ELSE}\\
&\qquad \textit{signal}(t)\leftarrow \textit{WAIT}\\
&\text{END IF}
\end{aligned} \tag{1}$$

In each training interval, the trading decision depends on the trading signal as follows.

- **BUY**: If we do not own the stock, then we buy the stock with all capital.
- **SELL**: If we hold the stock, then we sell all the stocks.
- **WAIT**: Do nothing.

In our method, a good strategy should possess that it has high return, and the sum of the positive returns in all transactions is higher than the absolute sum of negative returns. Besides, if there are two strategies with the same return, but one has more trading counts than the other, then we suggest that the former strategy may be able to make beneficial decisions. Accordingly, the fitness function for GEP in our method is defined as follows.

$$Fitness=(ROI_{cum}+\beta\times\frac{\sum ROI_{positive}}{100\times|\sum ROI_{negative}|})\times(1+\alpha\times T), \tag{2}$$

where $\alpha$ and $\beta$ represent the predefined weights, $T$ represents the total trading counts, $ROI_{cum}$ represents the cumulative return in the training interval, and $ROI_{positive}$ and $ROI_{negative}$ denote the positive return and the negative return in all transactions, respectively. If there is no transaction with negative return in the training interval, we set $|\sum ROI_{negative}|=0.01$.

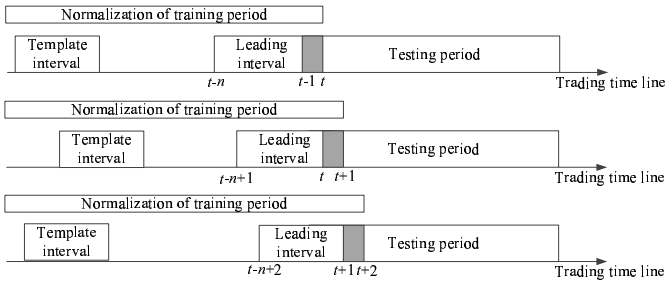


Figure 2. The trading day with the sliding window scheme. Here, $t$ and $n$ represent the trading day, represented by the gray block, and the length of template interval, respectively.

In general, the higher value of $\alpha$ is, the strategy with more trading counts is preferred. Similarly, if the higher value of $\beta$ is, the strategy with more positive returns is preferred.

## IV. Trading Strategies in the Testing Period

The data before the current trading day can be used as the training dataset. The generation of trading strategies in the testing period is operated in the sliding window matter, as shown in Figure 2. The *leading interval* is the time interval before the trading day. The length of the leading interval is denoted as $n$.

Our method for making trading decisions in testing period is given as follows.

1) **Set the parameters.**
   Set the value of $n$, which is the length of the leading interval. The values of $\alpha$ and $\beta$ in Eq. 2 for the fitness function in GEP are also set.
2) **Search for similar historical interval.**
   Search for sixty intervals, each of length $n$, from the historical data before the leading interval that the curves of investment return in the sixty historical interval are the most similar to the leading interval. The similarity measurement used here is the cosine distance, whose equation is given as follows.

$$Distance(V_1,V_2)=\frac{\sum_{i=1}^{n}(V_{1,i}\times V_{2,i})}{\sqrt{\sum_{i=1}^{n}V_{1,i}^2}\times\sqrt{\sum_{i=1}^{n}V_{2,i}^2}}, \tag{3}$$

   where $V_{1,i}$, $V_{2,i}$ and $n$ represent the $i$th element of the first ROI series $V_1$, the $i$th element of the second ROI series $V_2$, and the length of the ROI series, respectively. That is, the distance is equal to $\cos\theta$, where $\theta$ denotes the angle between the two vectors. Thus, the higher cosine distance of the two vectors have, the more similar they are.
3) **Find five non-overlapping template intervals in the strategy pool.**
   Each of the sixty similar intervals may not exactly correspond to one pre-trained interval in the strategy

pool, since the training intervals are obtained every 10 days. Thus, it is mapped to the nearest pre-trained interval. Finally, only five non-overlapping pre-trained intervals are filtered out from the sixty interval. Each of such interval is called a *template interval*, as shown in Figure 2.

4) **Decide the trading signal by vote.**
The trading strategies with positive returns for the five template intervals are fetched from the strategy pool. Then, a simple majority vote is performed for determining the final trading signal. To keep the stability of trading signal, we compute the *available ratio* ($\delta_A$), *buying ratio* ($\delta_B$) and *selling ratio* ($\delta_S$) in each trading day as follows.

$$\delta_A(t) = \frac{B_t + S_t}{B_t + S_t + W_t}, \tag{4}$$

$$\delta_B(t) = \frac{B_t}{B_t + S_t}, \tag{5}$$

$$\delta_S(t) = \frac{S_t}{B_t + S_t}, \tag{6}$$

where $t$, $B_t$, $S_t$ and $W_t$ represent the trading day, the numbers of BUY, SELL and WAIT signals, respectively. We also pre-set two parameters *available threshold* ($\gamma_A$) and *voting threshold* ($\gamma_V$). The trading signal of the trading day is finally decided as follows.

- **Buying**: If $\delta_A > \gamma_A$, $\delta_B > \gamma_V$ and we do not own the stock, then we buy the stock with all capital.
- **Selling**: If $\delta_A > \gamma_A$, $\delta_S > \gamma_V$ and we hold the stock, then we sell all of the stocks.
- **Waiting**: When both conditions of buying and selling are not satisfied, we do nothing.

5) **Check whether the testing period finishes.**
If the testing period does not finish, the current trading day is shifted to the next trading day and Steps 2 through 4 are executed iteratively.

6) **Compute the cumulative return.**
When the testing period finishes, the cumulative return is computed.

## V. Experimental Results

In this section, we will introduce the dataset used for experiments, the experimental results with performance comparison to previous results.

### A. Datasets

We use TAIEX as our investment target. Our dataset contains the basic information of TAIEX which are fetched from the Taiwan Economic Journal (TEJ) database. We assume that TAIEX is tradable. Our goal is to gain high and stable return by trading the daily close price of TAIEX. The close prices of TAIEX from 1995/1/5 to 2012/5/15 are shown in Figure 3. In our experiments, the training period is from 1995/1/5 to 2012/1/17. The testing period is from 2000/9/14 to 2012/1/17, where the close price of the starting day is very near to the ending day. Therefore, we can compare our result with the buy-and-hold strategy fairly.

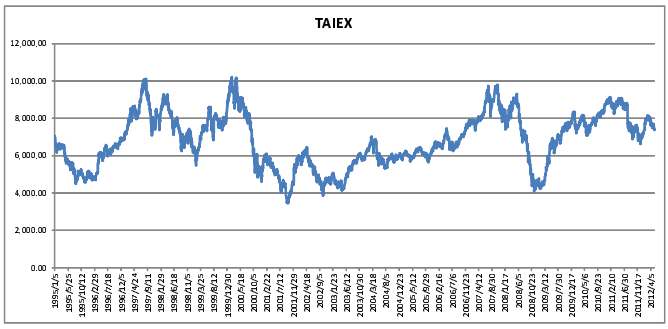


Figure 3. The close prices of TAIEX from 1995/1/5 to 2012/5/15.

Table I
THE PARAMETERS OF THE GENE EXPRESSION PROGRAMMING.

| Parameters | Content |
|---|---|
| Number of genes | 2 |
| Length of Head | 5 |
| Population size | 1000 |
| Number of generations | 200 |
| Selection method | roulette wheel |
| Mutation rate | 22.2% |
| IS transposition rate | 10% |
| RIS transposition rate | 10% |
| Gene transposition rate | 10% |
| 1-point recombination rate | 30% |
| 2-point recombination rate | 30% |
| Gene recombination rate | 10% |

### B. Experimental Results

The strategy pools are built for the training period , where only the data before the current trading day are used as training data. The lengths of the leading intervals, denoted as $n$, are 60, 90, 120, 180, and 270 days. The values of $\alpha$ and $\beta$ in Eq. 2 for the fitness function are set to 0.1 and 1, respectively. The parameters of the GEP for building the trading strategy pool is shown in Table I.

For each length $n$ of the training interval, we run five experiments to test the stability of the returns. In each experiment, $\gamma_A$ varies from 0.01 to 0.70 with step 0.01 and $\gamma_V$ varies from 0.50 to 0.89 with step 0.01, therefore there are $70 \times 40 = 2800$ simulations. Thus, we did 14000 simulations in total for each $n$.

To visualize the return distribution, the average cumulative return of each range of 0.05 in $\gamma_A$ and $\gamma_V$ is computed. The average cumulative returns and the standard deviations of training intervals with 60, 90, 120, 180 and 270 days are shown in Figures 4, 5, 6, 7 and 8, respectively. In each cell, the former and the latter represent the average cumulative return and the standard deviation, respectively. From these experiment results, we observe that the lower $\gamma_A$ combining with higher $\gamma_V$ can earn high return. The reason is that stronger signals may make trading decision correctly, especially the fewer *Buying* and *Selling* signals. The higher $\gamma_A$ combining with lower $\gamma_V$ also earns return, since the abundant trading signals usually make trading decision clearly. The situation is also satisfied in higher $\gamma_A$ combining with higher $\gamma_V$, but the trading counts decreases because the trading threshold becomes higher.

| $\gamma_A \setminus \gamma_V$ | 0.50 ~ 0.54 | 0.55 ~ 0.59 | 0.60 ~ 0.64 | 0.65 ~ 0.69 | 0.70 ~ 0.74 | 0.75 ~ 0.79 | 0.80 ~ 0.84 | 0.85~0.89 |
|---|---|---|---|---|---|---|---|---|
| 0.01 ~ 0.05 | -82.51% / 3.40% | -63.69% / 4.80% | -50.52% / 1.94% | -50.50% / 1.02% | -37.46% / 3.40% | -31.29% / 6.41% | -8.81% / 4.24% | 4.22% / 5.61% |
| 0.06 ~ 0.10 | -82.51% / 3.40% | -63.69% / 4.80% | -50.52% / 1.94% | -50.50% / 1.02% | -37.46% / 3.40% | -31.29% / 6.41% | -8.81% / 4.24% | 4.22% / 5.61% |
| 0.11 ~ 0.15 | -82.52% / 3.39% | -63.71% / 4.81% | -50.48% / 1.94% | -50.50% / 1.00% | -37.53% / 3.45% | -31.35% / 6.33% | -8.86% / 4.31% | 4.22% / 5.61% |
| 0.16 ~ 0.20 | -82.36% / 3.40% | -63.64% / 4.85% | -50.35% / 2.01% | -50.49% / 0.96% | -37.43% / 3.50% | -31.03% / 6.53% | -7.44% / 4.80% | 4.32% / 5.23% |
| 0.21 ~ 0.25 | -81.42% / 3.63% | -62.21% / 4.92% | -49.06% / 2.29% | -49.71% / 1.05% | -36.84% / 3.52% | -30.86% / 6.61% | -7.59% / 4.88% | 1.64% / 4.22% |
| 0.26 ~ 0.30 | -80.35% / 3.62% | -61.70% / 4.99% | -48.30% / 2.62% | -48.89% / 1.39% | -36.53% / 3.29% | -33.13% / 6.71% | -6.93% / 6.01% | -0.18% / 4.84% |
| 0.31 ~ 0.35 | -78.65% / 3.81% | -58.51% / 5.40% | -43.82% / 2.47% | -45.24% / 1.63% | -35.57% / 2.99% | -32.25% / 6.86% | -6.99% / 5.27% | 7.52% / 8.86% |
| 0.36 ~ 0.40 | -72.92% / 4.41% | -51.36% / 4.93% | -40.72% / 2.07% | -43.92% / 2.17% | -34.85% / 3.22% | -32.42% / 4.46% | -10.74% / 3.12% | 6.62% / 15.10% |
| 0.41 ~ 0.45 | -61.17% / 5.62% | -43.82% / 5.23% | -35.25% / 3.49% | -32.08% / 5.69% | -21.17% / 9.61% | -31.92% / 5.89% | -10.09% / 3.81% | 21.04% / 19.22% |
| 0.46 ~ 0.50 | -43.64% / 6.09% | -27.72% / 8.73% | -17.87% / 12.69% | -17.43% / 9.93% | -11.20% / 8.54% | -19.92% / 7.46% | 5.26% / 10.55% | 28.54% / 13.62% |
| 0.51 ~ 0.55 | -25.58% / 13.48% | -14.26% / 8.53% | -9.31% / 6.17% | -6.54% / 10.21% | -15.24% / 15.05% | -25.50% / 9.41% | 12.19% / 19.18% | 43.84% / 21.47% |
| 0.56 ~ 0.60 | 2.81% / 5.25% | 2.16% / 14.33% | -6.00% / 18.20% | -5.94% / 13.92% | -4.69% / 5.11% | 10.78% / 14.65% | 51.58% / 34.00% | 79.55% / 30.97% |
| 0.61 ~ 0.65 | 21.38% / 16.38% | 30.26% / 13.03% | 29.81% / 15.04% | 34.66% / 20.59% | 22.24% / 13.80% | 25.80% / 10.46% | 48.05% / 16.97% | 40.21% / 32.17% |
| 0.66 ~ 0.70 | 14.25% / 4.84% | 18.83% / 3.87% | 27.33% / 3.88% | 29.96% / 15.90% | 13.47% / 8.32% | 12.05% / 11.03% | 15.04% / 13.40% | 1.07% / 2.89% |

Figure 4. The average cumulative return and the standard deviation with training interval $n = 60$.

| $\gamma_A \setminus \gamma_V$ | 0.50 ~ 0.54 | 0.55 ~ 0.59 | 0.60 ~ 0.64 | 0.65 ~ 0.69 | 0.70 ~ 0.74 | 0.75 ~ 0.79 | 0.80 ~ 0.84 | 0.85~0.89 |
|---|---|---|---|---|---|---|---|---|
| 0.01 ~ 0.05 | -82.77% / 3.85% | -70.66% / 5.03% | -41.09% / 9.49% | -20.14% / 3.53% | -2.18% / 8.85% | 8.04% / 9.71% | 25.36% / 8.11% | 75.66% / 10.86% |
| 0.06 ~ 0.10 | -82.77% / 3.85% | -70.66% / 5.03% | -41.09% / 9.49% | -20.14% / 3.53% | -2.18% / 8.85% | 8.04% / 9.71% | 25.36% / 8.11% | 75.66% / 10.86% |
| 0.11 ~ 0.15 | -82.80% / 3.83% | -70.76% / 4.98% | -41.34% / 9.48% | -20.85% / 3.35% | -3.13% / 8.89% | 7.13% / 9.67% | 25.36% / 8.11% | 75.66% / 10.86% |
| 0.16 ~ 0.20 | -82.55% / 3.86% | -70.59% / 4.79% | -41.94% / 9.12% | -21.39% / 3.55% | -3.41% / 8.94% | 7.06% / 9.75% | 25.36% / 8.11% | 75.66% / 10.86% |
| 0.21 ~ 0.25 | -82.74% / 3.89% | -70.92% / 4.67% | -43.01% / 8.80% | -22.26% / 3.73% | -4.98% / 8.44% | 5.29% / 10.02% | 24.52% / 8.21% | 75.20% / 11.53% |
| 0.26 ~ 0.30 | -82.43% / 3.83% | -69.98% / 5.20% | -40.46% / 8.48% | -20.17% / 3.36% | -5.20% / 6.96% | -0.40% / 8.75% | 19.25% / 9.64% | 74.77% / 11.66% |
| 0.31 ~ 0.35 | -79.28% / 3.59% | -67.17% / 4.94% | -35.63% / 10.05% | -16.73% / 5.88% | -2.48% / 7.35% | -1.96% / 6.51% | 19.20% / 10.63% | 88.92% / 17.22% |
| 0.36 ~ 0.40 | -75.39% / 3.79% | -64.31% / 4.13% | -34.98% / 10.15% | -14.61% / 5.67% | 5.30% / 10.13% | 3.23% / 2.57% | 15.96% / 13.93% | 90.91% / 16.35% |
| 0.41 ~ 0.45 | -69.59% / 5.94% | -57.40% / 5.82% | -33.34% / 9.57% | -19.26% / 6.35% | 0.05% / 11.42% | 15.22% / 7.53% | 44.61% / 20.13% | 116.85% / 17.26% |
| 0.46 ~ 0.50 | -54.57% / 8.34% | -46.27% / 6.73% | -24.11% / 9.55% | -8.67% / 8.55% | 17.30% / 16.88% | 42.44% / 12.77% | 44.37% / 9.86% | 92.05% / 19.15% |
| 0.51 ~ 0.55 | -30.54% / 14.59% | -19.93% / 15.90% | 8.94% / 14.27% | 12.38% / 11.90% | 27.79% / 18.54% | 58.76% / 10.20% | 55.16% / 11.29% | 61.42% / 25.07% |
| 0.56 ~ 0.60 | 10.44% / 17.76% | 9.01% / 19.00% | 26.38% / 12.37% | 15.18% / 10.94% | 24.64% / 11.16% | 51.51% / 12.56% | 46.78% / 8.27% | 40.17% / 10.26% |
| 0.61 ~ 0.65 | 19.20% / 21.11% | 29.96% / 18.97% | 29.54% / 21.92% | 28.36% / 8.75% | 43.07% / 14.44% | 53.28% / 6.56% | 53.88% / 6.73% | 28.70% / 11.11% |
| 0.66 ~ 0.70 | 30.47% / 16.54% | 25.57% / 15.26% | 14.41% / 6.75% | 8.83% / 7.91% | 8.97% / 12.16% | 11.93% / 17.65% | 9.81% / 19.79% | 3.19% / 7.14% |

Figure 5. The average cumulative return and the standard deviation with training interval $n = 90$.

| $\gamma_A \setminus \gamma_V$ | 0.50 ~ 0.54 | 0.55 ~ 0.59 | 0.60 ~ 0.64 | 0.65 ~ 0.69 | 0.70 ~ 0.74 | 0.75 ~ 0.79 | 0.80 ~ 0.84 | 0.85~0.89 |
|---|---|---|---|---|---|---|---|---|
| 0.01 ~ 0.05 | -82.28% / 2.67% | -68.57% / 4.25% | -51.61% / 3.63% | -44.93% / 2.93% | -27.68% / 3.24% | -7.26% / 15.07% | 19.40% / 6.47% | 31.19% / 11.09% |
| 0.06 ~ 0.10 | -82.28% / 2.67% | -68.57% / 4.25% | -51.61% / 3.63% | -44.93% / 2.93% | -27.44% / 3.26% | -6.89% / 15.13% | 20.04% / 6.57% | 29.56% / 11.23% |
| 0.11 ~ 0.15 | -82.30% / 2.64% | -68.68% / 4.27% | -51.73% / 3.59% | -44.97% / 3.02% | -27.16% / 3.36% | -6.41% / 15.24% | 20.29% / 6.66% | 29.36% / 11.21% |
| 0.16 ~ 0.20 | -82.09% / 2.67% | -68.51% / 4.27% | -51.53% / 3.57% | -44.73% / 3.20% | -26.49% / 3.35% | -5.84% / 15.16% | 21.53% / 6.98% | 28.62% / 10.96% |
| 0.21 ~ 0.25 | -81.16% / 2.86% | -67.15% / 4.38% | -50.18% / 3.84% | -43.57% / 2.95% | -25.47% / 3.93% | -1.74% / 15.79% | 21.58% / 6.20% | 25.36% / 12.91% |
| 0.26 ~ 0.30 | -79.40% / 3.19% | -63.71% / 5.20% | -46.49% / 4.47% | -40.10% / 3.35% | -25.81% / 3.35% | -3.28% / 14.24% | 17.82% / 5.50% | 11.44% / 13.05% |
| 0.31 ~ 0.35 | -76.97% / 3.35% | -59.89% / 5.95% | -41.42% / 5.17% | -32.44% / 4.04% | -23.19% / 2.57% | -8.56% / 8.75% | 6.77% / 6.51% | 0.27% / 10.68% |
| 0.36 ~ 0.40 | -73.52% / 3.04% | -58.36% / 5.59% | -39.12% / 7.49% | -28.38% / 4.96% | -19.57% / 3.36% | -8.91% / 8.70% | 9.59% / 5.85% | 9.56% / 14.17% |
| 0.41 ~ 0.45 | -67.59% / 3.12% | -61.23% / 4.59% | -47.52% / 7.34% | -29.81% / 6.33% | -22.56% / 4.66% | -14.82% / 5.69% | 23.71% / 11.46% | 36.99% / 11.25% |
| 0.46 ~ 0.50 | -57.71% / 5.50% | -51.80% / 3.85% | -42.88% / 5.51% | -26.73% / 5.07% | -21.21% / 7.29% | 1.54% / 9.66% | 44.79% / 9.98% | 40.86% / 12.99% |
| 0.51 ~ 0.55 | -29.45% / 9.90% | -20.12% / 10.85% | -23.06% / 14.13% | -14.63% / 15.51% | -1.70% / 18.68% | 27.12% / 12.90% | 51.38% / 17.81% | 22.70% / 9.46% |
| 0.56 ~ 0.60 | 1.47% / 16.88% | 7.89% / 13.49% | 9.16% / 11.28% | 26.34% / 12.21% | 47.84% / 12.18% | 56.91% / 13.25% | 46.85% / 20.35% | 22.16% / 8.18% |
| 0.61 ~ 0.65 | 6.32% / 5.05% | 6.12% / 6.17% | -0.44% / 6.85% | 8.56% / 11.23% | 19.58% / 14.12% | 18.17% / 8.14% | 17.11% / 12.13% | 12.50% / 6.71% |
| 0.66 ~ 0.70 | -2.09% / 5.92% | 0.80% / 5.17% | 1.35% / 3.74% | 2.44% / 4.86% | 6.05% / 4.45% | 5.49% / 4.49% | 5.39% / 4.40% | 5.39% / 4.40% |

Figure 6. The average cumulative return and the standard deviation with training interval $n = 120$.

| $\gamma_A \setminus \gamma_V$ | 0.50 ~ 0.54 | 0.55 ~ 0.59 | 0.60 ~ 0.64 | 0.65 ~ 0.69 | 0.70 ~ 0.74 | 0.75 ~ 0.79 | 0.80 ~ 0.84 | 0.85~0.89 |
|---|---|---|---|---|---|---|---|---|
| 0.01 ~ 0.05 | -80.39% / 3.53% | -68.77% / 4.07% | -58.89% / 3.04% | -44.70% / 2.82% | -26.65% / 7.55% | 10.78% / 7.73% | 43.22% / 10.49% | 135.48% / 53.46% |
| 0.06 ~ 0.10 | -80.39% / 3.53% | -68.77% / 4.07% | -58.89% / 3.04% | -44.70% / 2.82% | -26.65% / 7.55% | 10.78% / 7.73% | 43.22% / 10.49% | 135.48% / 53.46% |
| 0.11 ~ 0.15 | -80.31% / 3.55% | -68.69% / 4.03% | -58.86% / 3.05% | -44.59% / 2.90% | -25.44% / 8.01% | 13.23% / 8.89% | 48.24% / 11.18% | 144.99% / 56.55% |
| 0.16 ~ 0.20 | -79.66% / 3.64% | -68.25% / 3.68% | -59.06% / 2.91% | -44.35% / 3.41% | -24.21% / 8.43% | 16.75% / 9.53% | 55.02% / 12.16% | 160.92% / 52.36% |
| 0.21 ~ 0.25 | -78.49% / 4.15% | -66.93% / 3.42% | -57.55% / 2.55% | -40.00% / 5.20% | -22.46% / 8.34% | 16.30% / 9.88% | 53.79% / 13.06% | 159.69% / 50.71% |
| 0.26 ~ 0.30 | -75.73% / 5.17% | -62.92% / 3.65% | -55.21% / 3.52% | -35.18% / 5.62% | -14.29% / 8.91% | 21.44% / 10.07% | 55.47% / 12.94% | 154.46% / 48.91% |
| 0.31 ~ 0.35 | -70.88% / 4.74% | -59.00% / 3.52% | -49.49% / 3.83% | -32.58% / 5.11% | -10.51% / 9.48% | 25.10% / 10.83% | 65.91% / 16.09% | 132.69% / 31.48% |
| 0.36 ~ 0.40 | -63.18% / 5.36% | -48.03% / 7.70% | -35.41% / 6.29% | -19.47% / 6.56% | 5.38% / 12.80% | 55.46% / 22.92% | 93.02% / 16.14% | 103.94% / 15.52% |
| 0.41 ~ 0.45 | -49.36% / 5.38% | -36.62% / 5.62% | -25.48% / 3.07% | -19.63% / 2.16% | 8.36% / 10.99% | 60.59% / 19.22% | 92.64% / 8.19% | 96.34% / 14.23% |
| 0.46 ~ 0.50 | -42.14% / 6.09% | -30.58% / 5.46% | -16.62% / 5.85% | -3.04% / 8.14% | 22.84% / 13.77% | 64.47% / 9.35% | 73.67% / 12.28% | 73.84% / 16.67% |
| 0.51 ~ 0.55 | -28.94% / 6.65% | -17.59% / 7.73% | -10.42% / 3.14% | 7.89% / 9.25% | 25.70% / 12.17% | 61.12% / 19.88% | 73.50% / 16.64% | 64.36% / 12.91% |
| 0.56 ~ 0.60 | 1.44% / 14.33% | 6.80% / 10.54% | 12.27% / 12.14% | 32.48% / 11.29% | 37.68% / 9.03% | 75.33% / 12.84% | 62.11% / 27.53% | 31.65% / 8.01% |
| 0.61 ~ 0.65 | 72.25% / 12.37% | 55.93% / 8.17% | 63.76% / 11.00% | 71.85% / 5.41% | 68.23% / 8.95% | 68.85% / 9.09% | 48.16% / 23.34% | 20.79% / 8.78% |
| 0.66 ~ 0.70 | 43.14% / 17.52% | 36.48% / 12.39% | 32.74% / 20.41% | 31.72% / 22.93% | 29.87% / 21.23% | 25.31% / 20.14% | 17.37% / 13.31% | 7.48% / 4.16% |

Figure 7. The average cumulative return and the standard deviation with training interval $n = 180$.

| $\gamma_A \setminus \gamma_V$ | 0.50 ~ 0.54 | 0.55 ~ 0.59 | 0.60 ~ 0.64 | 0.65 ~ 0.69 | 0.70 ~ 0.74 | 0.75 ~ 0.79 | 0.80 ~ 0.84 | 0.85~0.89 |
|---|---|---|---|---|---|---|---|---|
| 0.01 ~ 0.05 | -53.48% / 4.98% | -24.52% / 9.48% | -2.05% / 6.32% | 17.88% / 2.49% | 15.94% / 6.13% | 39.15% / 11.95% | 97.05% / 20.19% | 164.11% / 10.86% |
| 0.06 ~ 0.10 | -53.48% / 4.98% | -24.52% / 9.48% | -2.05% / 6.32% | 17.88% / 2.49% | 15.94% / 6.13% | 39.15% / 11.95% | 97.05% / 20.19% | 164.11% / 10.86% |
| 0.11 ~ 0.15 | -53.49% / 4.92% | -24.91% / 9.12% | -2.03% / 6.39% | 18.20% / 2.70% | 16.71% / 6.24% | 42.83% / 10.96% | 97.01% / 19.50% | 163.47% / 11.38% |
| 0.16 ~ 0.20 | -53.90% / 4.86% | -25.46% / 8.77% | -2.90% / 5.82% | 15.00% / 3.57% | 16.19% / 6.18% | 43.85% / 8.36% | 92.45% / 17.11% | 161.66% / 13.92% |
| 0.21 ~ 0.25 | -54.58% / 4.73% | -29.33% / 7.97% | -8.60% / 6.82% | 12.28% / 3.36% | 14.06% / 7.12% | 42.06% / 8.54% | 89.32% / 16.20% | 153.24% / 13.35% |
| 0.26 ~ 0.30 | -55.99% / 4.77% | -28.88% / 9.05% | -7.45% / 5.90% | 14.43% / 3.18% | 14.63% / 6.91% | 40.93% / 8.80% | 80.71% / 16.38% | 150.93% / 13.85% |
| 0.31 ~ 0.35 | -53.63% / 3.73% | -27.46% / 8.97% | -7.69% / 5.07% | 13.94% / 3.82% | 11.21% / 5.11% | 37.65% / 9.38% | 85.22% / 21.14% | 150.56% / 11.57% |
| 0.36 ~ 0.40 | -52.17% / 3.23% | -24.40% / 7.92% | -5.49% / 4.87% | 17.58% / 4.60% | 17.36% / 4.36% | 42.63% / 11.87% | 95.31% / 20.06% | 139.06% / 13.22% |
| 0.41 ~ 0.45 | -15.73% / 10.58% | 3.73% / 9.54% | 14.29% / 6.21% | 24.80% / 3.50% | 29.30% / 5.86% | 63.40% / 12.91% | 110.60% / 19.29% | 121.74% / 26.71% |
| 0.46 ~ 0.50 | 6.73% / 6.10% | 17.39% / 6.59% | 25.41% / 8.79% | 36.85% / 11.15% | 41.00% / 13.42% | 64.62% / 6.67% | 94.33% / 7.90% | 96.87% / 17.76% |
| 0.51 ~ 0.55 | 9.57% / 7.79% | 30.73% / 13.21% | 38.43% / 9.19% | 56.56% / 11.93% | 66.06% / 8.86% | 64.25% / 18.75% | 73.72% / 12.76% | 64.31% / 14.68% |
| 0.56 ~ 0.60 | 38.52% / 8.60% | 46.95% / 7.80% | 45.44% / 10.61% | 53.72% / 12.89% | 29.25% / 18.72% | 30.70% / 8.91% | 44.55% / 14.04% | 36.28% / 23.23% |
| 0.61 ~ 0.65 | 32.12% / 11.47% | 17.58% / 7.13% | 19.42% / 6.92% | 19.72% / 5.72% | 13.80% / 6.70% | 19.04% / 8.21% | 7.22% / 4.44% | 11.29% / 12.56% |
| 0.66 ~ 0.70 | 7.55% / 12.33% | 3.67% / 10.33% | -2.81% / 11.01% | -4.67% / 9.51% | -2.12% / 5.36% | 1.32% / 3.03% | 0.00% / 0.00% | 0.00% / 0.00% |

Figure 8. The average cumulative return and the standard deviation with training interval $n = 270$.

Table II
THE PARAMETERS OF THE BEST PERFORMANCE IN VARIOUS INTERVAL LENGTHS $n$ FROM 2000/9/14 TO 2012/1/17 BY TRADING TAIEX.

| Training interval | $\gamma_A$ | $\gamma_V$ | Avg. | Stdev. | Avg. / Stdev. |
|---|---|---|---|---|---|
| 60 | 0.57 | 0.84 | 153.75% | 137.16% | 112.10% |
| 90 | 0.41 | 0.88 | 139.47% | 19.81% | 704.22% |
| 120 | 0.54 | 0.81 | 81.20% | 45.00% | 180.43% |
| 180 | 0.21 | 0.88 | 236.25% | 115.96% | 203.74% |
| 270 | 0.19 | 0.88 | 187.63% | 71.66% | 261.84% |

## C. Comparison of Various Models

The parameters of the best performance in various interval lengths is shown in Table II. As one can see, the best performance in the 90-day training interval, which has the cumulative return 139.47%, is more stable than the other four best models. The highest cumulative return 236.25% occurring in the best 180-day training interval is obviously higher than the cumulative return 0.96% of the buy-and-hold strategy. The model with higher $\gamma_V$ usually earns high return, since this mechanism makes the trading decision correctly.

We also compare the *ROI* (return of investment), $ROI_{cum}$ (cumulative return) and $AR_{cum}$ (annualized return) with Jhou's method [8] on TAIEX from 2000/1/1 to 2009/12/31. The comparison is shown in Table III. We can see that the *ROIs* of our method are almost positive in every year except 2001. In addition, ROI of our method is more stable.

Table III
PERFORMANCE COMPARISON OF JHOU'S METHOD AND OUR METHOD WITH 180-DAY TRAINING INTERVAL FROM 2000/1/1 TO 2009/12/31.

| | Jhou's method with GP (training interval=730, testing interval =365 | | | Our method with GEP (training interval = 180, $\gamma_A$=0.21, $\gamma_V$=0.88) | | |
|---|---|---|---|---|---|---|
| year | *ROI* | $ROI_{cum}$ | $AR_{cum}$ | *ROI* | $ROI_{cum}$ | $AR_{cum}$ |
| 2000 | -1.34% | 6.75% | 3.32% | 0.00% | 0.00% | 0.00% |
| 2001 | -23.53% | -18.37% | -6.54% | -9.94% | -9.94% | -5.27% |
| 2002 | 47.26% | 20.21% | 4.71% | 30.43% | 17.04% | 4.83% |
| 2003 | -2.90% | 16.73% | 3.14% | 20.42% | 41.32% | 8.05% |
| 2004 | -0.14% | 16.57% | 2.59% | 2.36% | 44.27% | 6.82% |
| 2005 | -1.32% | 15.03% | 2.02% | 1.33% | 46.81% | 5.86% |
| 2006 | -4.40% | 9.97% | 1.20% | 21.13% | 80.42% | 7.89% |
| 2007 | 34.97% | 48.43% | 4.49% | 32.45% | 134.60% | 10.51% |
| 2008 | -15.69% | 25.14% | 2.27% | 0.00% | 134.60% | 9.28% |
| 2009 | 44.41% | 54.82% | 7.56% | 22.10% | 184.28% | 10.26% |

In Figure 9, we demonstrate that the trading time of our best 180-day training interval. As it shows, we usually buy

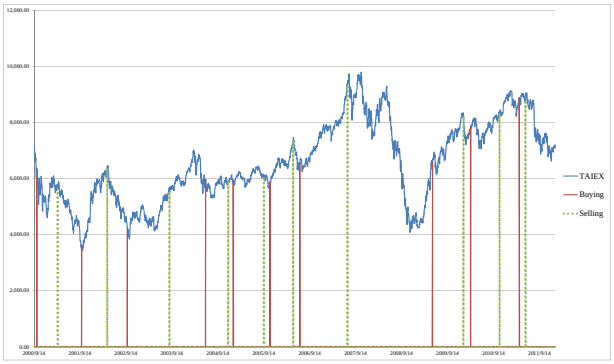


Figure 9. The transactions from 2000/9/14 to 2012/1/17 with 180-day training interval, $\gamma_A$ = 0.21 and $\gamma_V$ = 0.88. The solid red line and dotted green line represent each buying day and each selling day, respectively.

TAIEX at local valley and sell it at relatively high, such as 2nd, 3rd, 6th, 7th and 8th buying times and selling times. Even though the number of trading counts is not many, our method can still trade at good timing to earn positive return. It means that our method is profitable.

## VI. Conclusion

In the paper, we propose a method for investing in the TAIEX (Taiwan Stock Exchange Capitalization Weighted Stock Index). We utilize GEP to mine the trading strategies for the historical data in the training phase. And then, the trading decision mechanism based on the simple majority vote is adopted in the testing phase. The best cumulative return 236.25% occurs when the 180-day training interval is set with $\gamma_A$ = 0.21 and $\gamma_V$ = 0.88, which is higher than the cumulative return of the buy-and-hold strategy 0.96%.

From the results of our experiments, we find that the available threshold and the voting threshold are highly related to the return of investment. The model with higher voting threshold has fewer trading counts, and it usually gets better trading timings. The model with higher available threshold earns higher return than the model with low available threshold, since it makes trading decision more carefully.

In the future, some possible ways for improving investment performance may be done. The available threshold and the voting threshold may be changed dynamically to increase the possibility of profit. The feature selection may be adopted to mine the features highly related to the close price so that these features can be used to refine the trading strategy. To obtain effective strategies, the similarity measurement between the leading interval and template interval may include some classification techniques often used in time series data. Furthermore, the risk management and the portfolio combination may be considered for earning more stable return.

## References

[1] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," *AAAI Technical Report WS-94-03*, 1994.

[2] P. C. Chang, C. Y. Fan, and J. J. Lin, "Integrating a piecewise linear representation method with dynamic time warping system for stock trading decision making," *Fourth International Conference on Natural Computation*, Sept. 2008.

[3] H. H. Chen, C. B. Yang, and Y. H. Peng, "Genetic programming for the trading strategy of the mutual fund investment with Sortino ratio and mean variance model," *Proc. of the 15th Conference on Artificial Intelligence and Applications*, Hsinchu, Taiwan, Nov. 2010.

[4] C. Ferreira, "Gene expression programming: A new adaptive algorithm for solving problems," *Complex Systems*, Vol. 13, pp. 87–129, 2001.

[5] Y. M. Ha, S. Park, S. W. Kim, J. I. Won, and J. H. Yoon, "A stock recommendation exploiting rule discovery in stock databases," *Imformation and Software Technology*, Vol. 51, pp. 1140–1149, 2009.

[6] J. H. Holland, *Adaptation in natural and artificial systems: an introduction analysis with applications to biology, control, and artificial intelligence*. University of Michigan Press, 1975.

[7] C. M. Hsu, "A hybrid precedure for stock price prediction by integrating self-organizing map and genetic programming," *Expert Systems with Applications*, Vol. 38, pp. 14026–14036, 2011.

[8] S. M. Jhou, C. B. Yang, and H. H. Chen, "Taiwan stock forecasting with the genetic programming," *Proc. of the 16th Conference on Artificial Intelligence and Application (Domestic Track)*, Chungli, Taiwan, pp. 151–157, Nov. 2011.

[9] J. R. Koza, "Genetic programming: On the programming of computers by means of natural selectio," *MIT Press*, 1992.

[10] L. P. Ni, Z. W. Ni, and Y. Z. Gao, "Stock trend prediction based on fractal feature selection and support vector machine," *Expert Systems with Applications*, Vol. 38, pp. 5569–5576, 2011.

[11] T. J. Tsai, C. B. Yang, and Y. H. Peng, "Genetic algorithms for the investment of the mutual fund with global trend indicator," *Expert Systems with Applications*, Vol. 38(3), pp. 1697–1701, 2011.

[12] T. L. Wang and M. Wang, "Features extraction based on particle swarm optimization for high frequency financial data," *Granular Computing (GrC), 2011 IEEE International Conference on*, pp. 728–733, IEEE, 2011.