A Stock Pattern Recognition Algorithm Based on Neural Networks

Xinyu Guo
Institute of Computer Science & Technology
Peking University
100871 Beijing, China
guoxinyu@icst.pku.edu.cn

Xun Liang
Institute of Computer Science & Technology
Peking University
100871 Beijing, China
liangxun@icst.pku.edu.cn

Xiang Li
Institute of Computer Science & Technology
Peking University
100871 Beijing, China
lixiang@icst.pku.edu.cn

Abstract

Recent studies show that stock patterns might implicate useful information for stock price forecasting. The patterns underlying the price time series can not be discovered exhaustively by the pure man power in a limited time, thus the computer algorithm for stock price pattern recognition becomes more and more popular. Currently, there are mainly two kinds of stock price pattern recognition algorithms: the algorithm based on rule-matching and the algorithm based on template-matching. However, both of the two algorithms highly require the participation of domain experts, as well as their lacks of the learning ability. To solve these problems, the paper proposes a stock price pattern recognition approach based upon the artificial neural network. The experiment shows that the neural network can effectively learn the characteristics of the patterns, and accurately recognize the patterns.

1. Introduction

As an approach for stock investment, technical analysis has been widely-scrutinized by research communities, and the technical pattern analysis is regarded as one of the most important technical analysis approaches.

In the long-term stock analysis experience, stock analysts summarized many technical patterns beneficial for the investment decision-making, which can be classified into two categorizes: the continuation pattern and the reversal pattern. Continuation pattern indicates that the stock price is going to keep its current movement trend; while the reversal pattern indicates that the stock price will move to the opposite trend. 63 important technical patterns are detailed in [1]. In this paper, we choose 18 typical technical patterns as the research target, including 10 continuation patterns and 8 reversal patterns. Figure 1(a) and Figure 1(b) show one typical continuation pattern and one reversal pattern respectively.

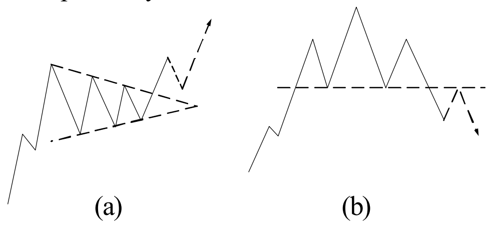

Figure 1. (a) A typical continuation pattern: ascending symmetrical triangle. (b) A typical reversal pattern: head and shoulders tops.

With the constant enlargement of the securities exchange market, it is unrealistic for the finance practitioners to discover patterns from thousands of hundreds of stocks in time without any professional aids. Scientists begin to design computer-based algorithms for technical pattern recognition [2, 3, 4, 5, 6, and 7]. The technical pattern recognition algorithm can mainly be classified into two categories, one is the rule-based algorithm [2, 5, 6, 7], and the other is template-based algorithm [3, 4]. Nonetheless, either of these two categories has to design a specific rule or template for each pattern, which requires highly professional skills as well as involves considerable risks thus posing an enormous challenge for technical analysts.

For the past several years, neural network has been going through an increment in popularity within stock market analysis. One typical illustration is the study conducted in [8], where a recognition algorithm for triangle patterns based upon a recurrent neural network was introduced. Nevertheless, it seems computationally expensive since the method needs to manipulate each time point within the series. In this work, we also propose an approach relying on neural network whereas the most noticeable difference, compared to the work done in [8], lies in that the inputs of the network do not cover every time point in the series. On the contrary, a segmentation process is adopted in this work to first transform the original time series into a sequence of trend segments and corresponding features, with each of the features calculated in terms of the price at the last time point within the segment. Eventually, this sequence of features, instead of the whole time series, is designated as part of the inputs of the network, which in comparison with [8], not only is able to reduce the calculation expense but also enable the alteration of time granularity for stock patterns by adjusting the length of the segments.

The paper is arranged as follows: Section 2 details the feature extraction process. The pattern recognition algorithm using neural network is discussed in section 3. Section 4 covers the experimental results and a conclusion is given in section 5.

2. Feature extraction

The stock price time series can be denoted by a curve within a certain plane. As given by Figure 2, where the x-coordinate represents the trading days while the y-coordinate the closing prices.

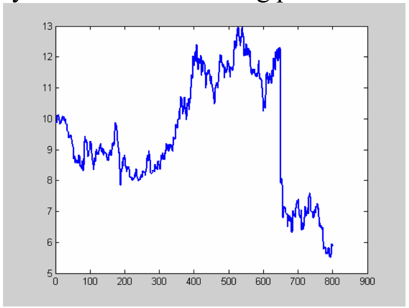

Figure 2. The closing price of YanJing Beer on a daily basis from July 25th, 2002 to November 23rd, 2005.

Because stock price time series contains a large number of time points, it is a time-consuming job to analyze the patterns from the time series directly. Therefore, a simplified but efficient method should be used to represent the time series. In this paper, we adopt the segmentation process to simplify it, as [2] and [9] do.

There are three methods to realize the trend segmentation, including sliding windows, top-down and bottom-up [10]. Suppose the stock price time series contains T time points, with the average length for segments as k ($1 < k < T$), thus the running time for both the sliding windows and the bottom-up algorithm is $O(T)$, while the top-down is $O(T^2)$. In this work, we choose the bottom-top approach to conduct the time series segmentation, with the specific steps as follows.

1) Put the data at both the $(2i-1)$th and the $2i$th time points into one segment, thus the original time series is divided into $T/2$ segments.

2) Calculate the cost of merging two adjacent segments together, namely the merging cost. Here we define the segment cost as the sum of all the squared distances between the data points within this segment and the corresponding line.

3) Merge the two adjacent segments together whose merging yields the smallest cost.

4) If the current segment number is greater than T/K, go to 3), otherwise 5).

5) Connect the neighbor segment in a head-to-tail manner, forming a continuous zigzag line.

6) Merge the two adjacent segments together if they share the same trend.

7) Delete those segments with tiny price oscillation.

8) If the ith segment has an upward trend while the $(i+1)$th has a downward trend, move the right end of the former (i.e. the left end of the latter) to the maximum data point within the two segments, and vice versa.

9) If the segmentation result changes during step 6), 7) or 8), go to 6), otherwise 10).

10) Return the sequences of trend segments.

The corresponding segmenting results, given that k=26 and k=5, are visualized in (a), (b) of Figure 3, respectively.

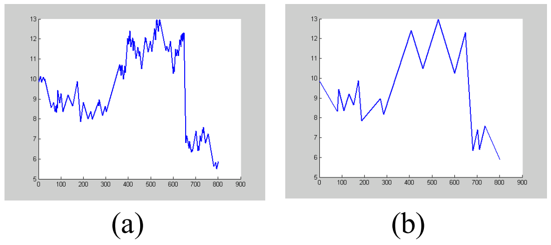

Figure 3. The segmentation results. (a) The segmentation result when k = 5 with 111 segments in total; (b) The segmentation result when k = 26 with 19 segments in total.

A technical pattern is composed of several continuous trend segments, based on which several samples can be extracted by a certain length of time window. For example, suppose a time series with the length T is divided into m trend segments, therefore the segment series can be denoted as $(s_1, s_2, \dots, s_m)$, with $s_i (1 \le i \le m)$ representing the ith segment. When the length of the time window is w, we can extract m-w+1 samples from this sequence of segments, denoted as $(s_1, s_2, \dots, s_w)$, $(s_2, s_3, \dots, s_{w+1})$, …, $(s_{m-w+1}, s_{m-w+2}, \dots, s_m)$.

As shown in Figure 4, assuming that the length of the time window is 7, we mark the end points of the segments sequentially as 0, 1, 2, …, 7, and make these 7 segments into a sample with 8 corresponding price value denoted as y_0, y_1, y_2, … , y_8. Generally, the movement range of the price varies according to the stock. In order to eliminate the influences caused by this variance, it is necessary to normalize the price into a uniform interval ([0, 1] for instance) [8], or to acquire the relative price value by calculating the proportion. The latter approach is adopted here in this paper.

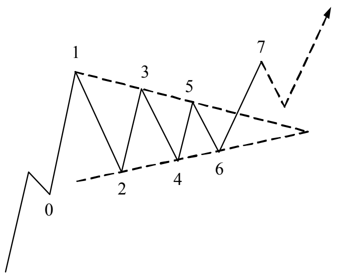

Figure 4. Marking the starting and ending points of each segment.

We define 8 features as follows for the sample with a length of 7.

$$p_1 = \begin{cases} 1, & \text{if } y_1 - y_0 \geq 0 \\ -1, & \text{if } y_1 - y_0 < 0 \end{cases} , \quad (1)$$

$$p_i = \frac{y_i - y_{i-1}}{y_{i-1} - y_{i-2}}, \quad i = 2,3,...,7 , \quad (2)$$

$$p_8 = \frac{y_7 - y_1}{|y_2 - y_1|}, \quad (3)$$

3. Pattern recognition using a classification neural network

3.1. Algorithm

In this paper we investigate the 18 kinds of typical technical patterns. Opposed to the aforementioned [8], in this work we transform the recognition into a process of classification using a three-layer feedforward neural network, whose inputs are the sample features defined in section 2.

A three-layer feedforward neural network is typically composed of one input layer, one output layer and one hidden layers. In the input layer, each neuron corresponds to a feature; while in the output layer, each neuron corresponds to a predefined pattern.

The first step to carry out the classification is to train the neural network with a group of training samples, which can be characterized as having expected outputs provided in them. What is worth mentioning is that every training sample belongs to a certain predefined pattern. Then we can use the testing samples to test the performance of the trained network. The best situation is that once a certain sample is input into the network, the output will be a vector with all elements as zero only except the one corresponding to the pattern that the sample belongs to. Nonetheless, due to the existence of classification errors and the fact that some testing samples don't belong to any of the 18 predefined patterns; some samples can not get exactly the expected output.

Here we try to decide whether a sample belongs to a certain pattern via the following method. We define two positive numbers h_1 and h_2, which are both very close to 0 (e.g. h_1=0.005, h_2=0.01), and we name them recognition thresholds. Regarding a certain input sample, and the corresponding output vector is (a_1, a_2, …, a_M), where M denotes the number of neurons in the output layer, then this sample should be categorized into the ith pattern only if $|a_i - 1| \leq h_1$ $(1 \leq i \leq M)$ and at the same time $a_j \leq h_2$ $(1 \leq j \leq M, j \neq i)$, otherwise, it is considered not belong to any pattern.

3.2. Metrics for the pattern recognition experiments

First of all, the most important metric for classification is the precision. Second, due to the fact that there are some samples in the testing set that can not be categorized into any predefined pattern, the ability of the neural network to distinguish such kind of samples out of the other samples is also considered to be of great significance. In order to evaluate the classification performance of the neural network, we define seven metrics as follows.

1) Classification precision. This term refers to the proportion of accurately-recognized patterns compared to all the samples that can be categorized as some patterns by the network, denoted as pr_c.

2) Classification precision for continuation (reversal) pattern. This term refers to the proportion of accurately-recognized samples as continuation (reversal) patterns compared to all the samples which are categorized by the network as continuation (reversal) patterns, denoted as cpr_c (rpr_c.).

3) Isolation precision. This term refers to the proportion of accurately-recognized samples (either belonging to some predefined pattern or not) compared to all the testing samples, denoted as pr_i.

4) Recall. This term refers to the proportion of accurately-recognized patterns compared to all the samples, contained in the testing set, which are of some patterns, denoted as rc.

5) Recall for continuation (reversal) pattern. This term refers to the proportion of accurately-recognized samples as continuation (reversal) patterns compared to all the samples, in the testing set, which are of continuation (reversal) patterns, denoted as crc (rrc).

4. Experiments

We chose 2029 samples out of 508 stocks from Shanghai Stock Exchange, which include 593 continuation patterns and 1436 reversal patterns, as training samples. At the mean time, we also extract 4937 samples as the testing samples out of 155 stocks from Shenzhen Stock Exchange within the same time-interval, there are 54 continuation patterns, 270 reversal patterns and 4613 belong to neither of the two.

4.1. Network training

In this work, the maximum number of segments in one sample is 7 with the feature number as 8. The input samples have to be first normalized before can be put into the network for training, by scaling all the feature values of each sample into the range of [-1, 1]. Each of the 18 neurons in the output layer of the network corresponds to a certain pattern, and the network is denoted as NN_A here. What's more, when the network is tested by the testing set, it is also necessary to normalize the testing samples.

The second lines of Table 1 and Table 2 show the network architecture and the testing result for NN_A respectively. h_1 and h_2 are the recognition thresholds as defined in the last section.

4.2. Feature discretization

Table 2 indicates that the classification precision and recall of NN_A is not satisfactory, which is probably due to the large range the sample features have stretched, while the amount of the training sample available is limited, thus lead to a subsequent overfitting and less generalization of the network. In response to that, we keep on analyzing the essence of the patterns, and then we discover that the pattern is mainly determined by the relative positions of the starting and ending points within each segment. We can describe the correlation between points by using a finite number of state indicators. For instance, in the context of two points, we can use 0 to indicate that the latter is lower than the former, 1 the latter is more or less the same height as the former, and 2 the latter is higher than the former. In this way the discrete feature values can be realized, namely the feature discretization. With these discrete features, we construct a new network using the approach adopted when building the NN_A, denote it as NN_B.

The third lines of Table 1 and Table 2 show the network architecture and the testing result for NN_B respectively. Obviously, the enhancement in generalization due to the feature discretization improves the classification precision and recall to a great extent.

4.3. Grouping training

As shown in Table 2, the recall for continuation pattern is rather low no matter whether the feature discretization is adopted or not. The main reason is that there exist a large number of reversal patterns containing fewer segments in the training set, which disturbs the learning process for continuation patterns. Confronting this situation, we first divide the training sample into 2 parts, the continuation patterns and the reversal patterns, and then use the two groups to train two feedforward networks, which are denoted as NN_C and NN_D. Since the patterns being investigated in this work contain 10 continuation patterns and 8 reversal ones, the numbers of the output neurons for NN_C and NN_D are 10 and 8 respectively. When a testing sample has been put into the networks for testing, both the categorization results of the two networks should be taken into consideration to produce the final recognition; specifically, if the NN_C categorizes it into continuation pattern while NN_D categorizes it into reversal pattern, the sample is considered to be classified into continuation pattern, whereas if NN_C outputs neither pattern for that sample, the output of NN_D is adopted as the final recognition.

The fourth and the fifth lines of Table 1 show the network architectures for NN_C and NN_D respectively. The fourth line of Table 2 shows the testing result for NN_C and NN_D.

The experimental results illustrate that the implementation of grouping training greatly enhances the recall for continuation pattern though decreases a little bit the classification precision for it, and the other metrics are no less than the NN_B network. Therefore, we regard the NN_C combining NN_D network as a better approach.

Table 1. The network architectures for different networks.

network	number of input neurons	number of hidden neurons	umber of output neurons	h_1	h_2
NN_A	8	200	18	0.27	0.8
NN_B	8	200	18	0.08	0.03
NN_C	8	200	10	0.01	0.01
NN_D	6	200	8	0.01	0.01

Table 2. The testing results for different networks.

network	pr_c	cpr_c	rpr_c	pr_i	rc	crc	rrc
NN_A	0.524	0.375	0.528	0.887	0.568	0.058	0.676
NN_B	0.983	1	0.983	0.981	0.840	0.167	0.913
NN_C+NN_D	0.963	0.796	0.993	0.995	0.951	0.722	0.996

5. Conclusion

With the fast enlargement of the stock market in their scales, it seems impossible to discover the underlying technical patterns by pure human power. The implementation of automatic pattern recognition using computer technologies serves as an effective alternative.

Searching for patterns from the original price time series is a time-consuming process and thus is not applicable for real-time pattern recognition. In this paper, we propose a efficient and effective approach to represent the stock price time series, which is adopted by patterns theory in technical analysis. The stock price time series is reduced to multiple line segments, from which we extract the pattern features, and the feedforward neural network is used for the pattern recognition. According to the empirical results, to separate the continuation and reversal patterns, by using two networks to conduct the recognitions individually, has observably increased the classification precision as well as the recall.

Within the current literature of technical analysis, generally the scale of the pattern, namely the time span, has always been considered as an important criterion to identify the pattern. Therefore, we will employ the time span factor in our classification model based upon artificial neural networks in our subsequent work, in order to enhance the accuracy and reliability of the pattern recognition.

References

[1] N. Bulkowski. *Encyclopedia of Chart Patterns*, 2nd Edition. John Wiley and Sons, 2005.
[2] C. L. Osler, and P. H. K. Chang. Head and Shoulders: Not Just a Flaky Pattern. Staff Report No.4, Federal Reserve Bank of New York.
[3]W. Leigh, N. Paz, and R. Purvis. Market timing: a test of a charting heuristic. *Economics Letters*, 77(1)(2002), 55-63.
[4] W. Leigh, N. Modani, and R. Hightower. A Computational Implementation of Stock Charting: Abrupt Volume Increase As Signal for Movement in New York Stock Exchange Composite Index. *Decision Support Systems*, 37(4)(2004), 515-530.
[5] A. Lo, H. Mamaysky, and J. Wang. Foundations of Technical Analysis: Computational Algorithms, Statistical Inference, and Empirical Implementation. *Journal of Finance*, 55(4)(2000), 1705-1765.
[6] S. C. Suh , D. Li, and J. Gao. A Novel Chart Pattern Recognition Approach: A Case Study on Cup with Handle. In the *Artificial Neural Network in Engineering Conference*, St. Louis, Missouri, 2004.
[7] S. Anand, W. N.Chin, and S. C. Khoo. Chart Patterns on Price History. *Proc. of ACM SIGPLAN Int. Conf. on Functional Programming*, 134-145. Florence, Italy. 2001.
[8] K. Kamijo, and T. Tanigawa. Stock Price Pattern Recognition: A Recurrent Neural Network Approach. *Proc. of the Int. Joint Conf. on Neural Networks*, vol. 1, 1990, 215-221.
[9] F. Chung, T. Fu, V. Ng, and R. Luk. An Evolutionary Approach to Pattern-based Time Series Segmentation. *IEEE Trans. on Evolutionary Computation*. 8(5)(2004), 471-489.
[10] E. Keogh, S. Chu, D. Hart, and M. Pazzani. An Online Algorithm for Segmenting Time Series. *Proc. of IEEE Int. Conf. on Data Mining*, 2001, 289-296.