Automatically Recognizing Stock Patterns Using RPCL Neural Networks

Xinyu Guo Xun Liang Nan Li

Institute of Computer Science and Technology, Peking University, Beijing 100871, P. R. China

Abstract

Stock patterns are those that occur frequently in stock time series, containing valuable forecasting information. In this paper, an approach to extract patterns and features from stock price time series is introduced. Thereafter, we employ two ANN-based methods to conduct clustering analyses upon the extracted samples, which are the self-organizing map (SOM) and the competitive learning. Besides, and we introduce an improved version of the rival penalized competitive learning (RPCL), and furthermore conduct a comparative study between the clustering performances of the improved RPCL and the SOM. Experimental results show that a better clustering performance can be achieved by the former.

Keywords: Competitive learning, Feed-forward neural network, Pattern analysis, Self-organizing map, Time series analysis.

1. Introduction

Within the current literature, there are two classic approaches for stock analysis, the fundamental analysis and the technical analysis. The former forecasts the tendency of stock price movement in light of the various elements that have an influence on the supply-and-demand correlation, while the latter does it via inspecting the market behaviors through graphs, charts as well as technical indicators. As yet, technical analysis has been comprehensively studied. The most frequently-applied theories in this regard include the Dow theory, the Elliot wave theory, the price volume theory, the K-line theory, the pattern theory and so forth, the majority of which combine chart analysis with technical indicator analysis in order to predict the moving trend of the stock price. Pattern analysis, an important branch of technical analysis, probes into the comparison between the strengths of the longs and shorts indicated by the stock curves. In a nutshell, pattern analysis is aimed at discovering both the undergoing and would-be trends in stock price with the aid of various patterns displayed by the corresponding curves. By investigating into the historical stock prices, pattern analysis serves to be suggestive and advisory to support investors to forecast stock price movements.

As for the technical patterns in pattern analysis, they refer to those patterns recurrently exhibited in stock price curves. Technical patterns hold their own importance in financial forecasting. [1] indicate the efficiency of one specific technical pattern, namely the head and shoulders, for forecasting the foreign exchange rate. [2] and [3] demonstrate that the bull flag technical pattern, exhibited in both the stock price and the trading volume time series, is able to generate trading rules that imply higher profits compared to stochastic trading strategies. Besides, it is claimed in [4] that some technical patterns can provide additional information to forecast stock prices. One predominant type of technical pattern is the candlestick chart using candlestick-like figures to represent four important prices within a trading day, namely the opening, the closing, the highest and the lowest. In this paper, we concentrate on the closing price time series, which is divided into multiple trend segments, each of which is denoted by a line segment indicating the basic trend of the stock price. An uptrend implies a rise in price whereas a downtrend a decrease.

The highly computational capabilities derived from computer technologies have enabled the design of computer-based algorithms for pattern analysis [1]-[8]. In the meanwhile, neural network has been going through an increment in popularity within stock market analysis [9]. In this work, we propose two clustering approaches relying on neural networks to analyze the patterns in stock price time series. The most noticeable difference, compared to the work done in [9], lies in that the inputs of the networks do not cover every time point in the series. On the contrary, a segmentation process, which is also proposed by [1], is adopted in this work to first transform the original time series into a sequence of trend segments and features. Eventually, this sequence of features, instead of the whole time series, is designated as part of the inputs of the network. This not only reduces the calculation expense but also enables the alteration of time granularity by adjusting the length of the segments.

ISBN: 978-90-78677-04-8

As mentioned before, technical patterns are characteristic of being recurrent. Assuming that those non-predetermined frequently-occurred patterns are some unknown clusters, clustering analysis can therefore be implemented. Two classic approaches include the self-organizing map (SOM) and competitive learning, both achieving the clustering process via a competition among several neurons for the current object. We investigate in detail the clustering performance of SOM as well as a variation of RPCL [11] in this work and the inputs of the network are feature sequence extracted from the smoothed trend segment sequence. Basically, the best performance can be achieved when the number of output units equals the number of clusters within the samples' feature space. However, it is usually an impossible task to foreknow the number of the clusters [10]-[13]. In [10], the number of output units is progressively curtailed by combining them with similar weight vectors. Nevertheless, the constant length of the sliding window, adopted in [10], vastly constricts the time-span for samples, and in the meantime, numerous redundant samples are generated due to that the window slides only one trading day forward each time. In this paper however, time-span varies among samples, thus leading to a comparatively low coupling between each.

The rest of paper is organized as follows. In section 2, the pattern theory and the trend segmentation are briefly described. Clustering analyses conducted upon the segmented samples are presented in section 3. Section 4 details the experiments and section 5 concludes this paper.

2. Pattern Theory and Trend Segmentation

2.1. Pattern theory in a nutshell

Pattern theory tends to dig into various shapes underlying those stock price curves, although not every shape is able to be used to carry out predictions. A number of beneficial technical patterns have been discovered by stock analysts, and these patterns could be categorized principally into two types: continuation pattern and reversal pattern. Continuation pattern indicates that stock price is going to keep its current movement trend thus the original balance will be maintained whereas the reversal pattern, on the contrary, implicates that the current balance will be violated and an opposite trend will appear. In terms of movement trend, continuation pattern can be further classified into the ascending and the descending continuations while reversal pattern further categorized into the top and the bottom reversals. A detailed investigation of as many as 63 important technical patterns, including the chart patterns as well as the event patterns, was made in [14]. Technical patterns reflect the market behavior [14], thus serving to be the footprints following which those informed investors can be able to enhance their profits. Fig. 1 visualizes ascending symmetrical triangle and head and shoulders top, two technical patterns that have already been studied.

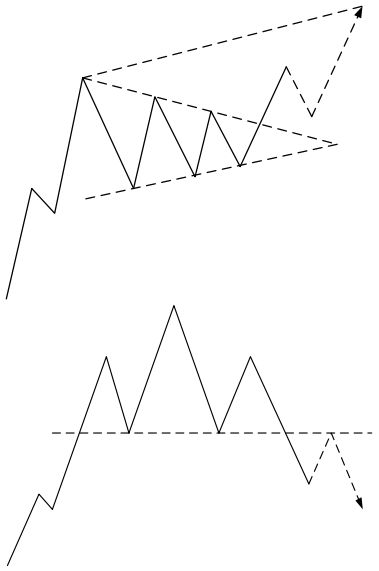

Fig. 1: Two technical patterns, with (a) ascending symmetrical triangle and (b) head and shoulders top.

2.2. Trend segmentation and feature extraction

For the sake of the efficiency and simplicity, trend segmentation is usually conducted upon the stock price time series. Generally, a stock price time series comprises of several coarse trends, each of which might further contain several fine trends. Discovering all the finest trends is laborious and somewhat redundant. Therefore, trend segmentation is employed in order to smooth off the relatively unimportant fine trends while keep reflecting the general coarse trends in large [1, 4, 15].

Segmentation involves first dividing the time series into multiple segments, according to some optimization rules, and then connecting each in a head-to-tail manner. Three approaches are usually employed, namely the sliding window method, the top-down method and the bottom-up method [16]. In this paper, we take the third method as the segmentation approach.

Suppose that we have T time points within the time series, with the average length for each segment l ($1<l<T$) time intervals. The bottom-up approach refers to a process that first a value is assigned to l, then the time series is divided into $\frac{T}{2}$ segments by grouping the (2i-1)th and the 2ith (i=1, 2,…, T/2) time points into one segment. Afterwards, calculation of the merging costs with its two neighbors for each segment is carried out. Two adjacent segments whose merging yields smaller cost are merged and new merging costs for the new segment is recalculated. This process is repeated until the number of the segments falls no more than $\frac{T}{l}$.

Feature extraction serves to be an important step after trend segmentation. After it has been transformed into multiple trend segments, the time series can be defined uniquely by the starting and ending points of all these segments. Suppose that a time series with the length T is divided into m trend segments, denoted by (s_1, s_2, …, s_m), with s_i ($1\le i\le m$) representing the ith segment. The length of time window is set to w, thus (m-w+1) samples can be extracted from this sequence of segments, as (s_1, s_2, …, s_w), (s_2, s_3, …, s_{w+1}), … , (s_{m-w+1}, s_{m-w+2}, …, s_m).

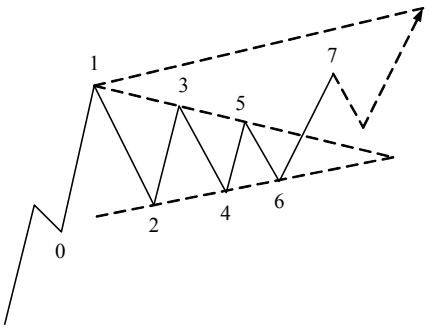

Fig. 2: Marking the starting and ending points of each segment within a sample.

Assuming that w is 7, Fig. 2 visualizes a sample with 7 segments. The conjunction time points are denoted as y_0, y_1, y_2, …, y_7. We define 8 features as follows for the sample with a length 7 as

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

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

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

where P_1 denotes the initial trend of the pattern, with a value 1 representing upward while -1 downward; P_2~P_7 indicate the proportions between the price spans of two adjacent segments; P_8 implies the correlation between the continuation pattern's breaking point and the first segment's right point. Thus we can use (P_1, P_2, …, P_8) to represent this sample.

3. Clustering Analysis for Stock Patterns

Due to its learning abilities, neural network has been widely implemented into clustering analysis, two typical approaches including self-organizing map (SOM) and competitive learning algorithms. A competitive learning network, which works in a winner-takes-all fashion, is characteristic of having only one neuron in the output vector as 1 whereas all the others 0. SOM network is a subtype of competitive learning network, differing from the other subtypes as introducing the concept of neighborhood. Nonetheless, most competitive learning networks, including SOM, suffer from the difficulty to choose the number of the output neurons, denoted as k. The optimal performance is only achieved when k is equivalent to the number of clusters within the sample space.

In response to this, L. Xu et al. introduced in [11] a heuristic adaptation of competitive learning, namely the rival penalized competitive learning (RPCL), which moves the rival neuron away from the current object (i.e. the input vector), while moving the winning neuron closer to the current object. The rival of the winning neuron in RPCL refers to the neuron whose weighted distance to the current object is only greater than the winning neuron while smaller than all the other neurons. One competitive learning epoch in RPCL contains two prime steps.

- A sample $\vec{x}$ is selected from the training sample set, and the output vector for this sample can be calculated by the rules

$$u_i = \begin{cases} 1, & \text{if } i = c, \text{ and } \gamma_c \left\| \vec{x} - \vec{w}_c \right\|^2 = \min_j \gamma_j \left\| \vec{x} - \vec{w}_j \right\|^2, \\ -1, & \text{if } i = r, \text{ and } \gamma_r \left\| \vec{x} - \vec{w}_r \right\|^2 = \min_{j \ne c} \gamma_j \left\| \vec{x} - \vec{w}_j \right\|^2, \\ 0, & \text{else.} \end{cases} \quad (4)$$

where u_i (i = 1, 2, …, k) represents the ith neuron of the output vector; c represents the serial number of the winning neuron while r the rival; $\vec{w}_j$ represents the weight vector of the jth neuron; $\gamma_j = \frac{n_j}{\sum_{i=1}^{k} n_i}$ while n_i denotes the cumulative number of winning times for the ith neuron.

- Secondly update the weight vector $\vec{w}_i$ for each neuron using

$$\Delta\vec{w}_i = \begin{cases} \alpha_c(\vec{x} - \vec{w}_i), & \text{if } u_i = 1, \\ -\alpha_r(\vec{x} - \vec{w}_i), & \text{if } u_i = -1, \\ 0, \text{ else.} \end{cases} \quad (5)$$

where α_c represents the learning rate of the winning neuron while α_r that of the rival's. For the purpose of learning stability, α_r should be much smaller than α_c (e.g. α_c=0.05, α_r=0.002). This is primarily because if α_r is assigned a rather big value, the rival will be quickly expelled out of the cluster area, which leads to an undesirable consequence that only quite a few clustering centers will be left.

What can be directly obtained by the RPCL is that every cluster contains only one cluster center, which is calculated based on the weight vector of the corresponding output neuron of the network, whereas all the redundant neurons are isolated away from all the clusters. The clustering result using RPCL algorithm when k is set to 5 is visualized in Fig. 3, where it's clear to notice that one neuron has been isolated away from all the clusters.

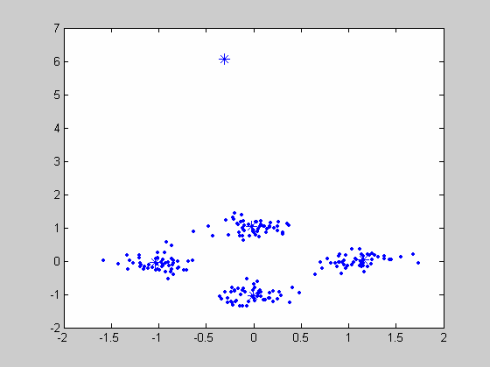

Fig. 3: The clustering result using RPCL algorithm when k is set to 5. Note that the neuron with the weight vector (-0.3, 6) has been isolated from the cluster area.

Therefore, RPCL serves as an efficient approach to solve the problem caused when k is set greater than the number of clusters. Nonetheless, when the value of k is initialized way too large, RPCL is considered not to be sufficient anymore. As illustrated by Fig. 4, when k is set to 6, RPCL is as incompetent as normal competitive learning algorithms. One primary reason for this is that when there are multiple redundant neurons, each of them is accordingly given less expelling force. One alternative to solve this problem is to increase the value of α_r (increase it to 0.02 for instance) thus increasing the moving speed of the rival neurons. However, such solution also suffers from an unstable learning process, since the mutual expelling processes among neurons tend to induce the deviation of cluster centers from the cluster means. The clustering result of RPCL when k=6 and α_r=0.02, is shown in Fig. 5.

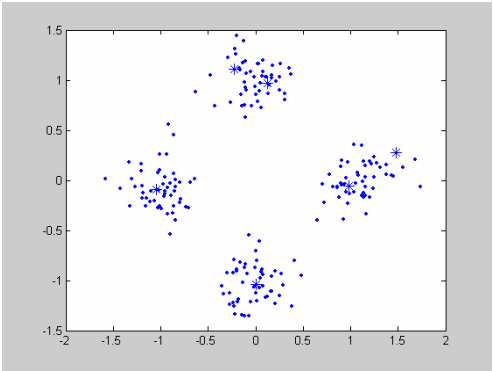

Fig. 4: The clustering result using RPCL algorithm when k is set to 6. Note that due to the insufficiency of RPCL, the redundant neurons are not completely expelled from the cluster area while still stay in the neighborhood instead.

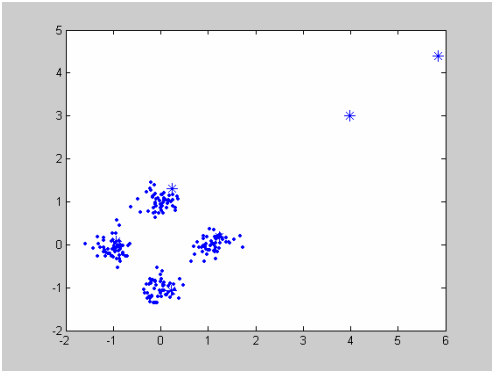

Fig. 5: The clustering result using RPCL algorithm when k is set to 6 and α_r is increased to 0.02. Note that due to the increment of α_r, a side effect has come along that three clustering centers have deviated from the cluster means, which are (0, -1), (0, 1) and (1, 0).

For the sake of learning stability, we implement different values of α_r for different training epochs. We gradually decrease its value and assign it to 0 eventually, which constitutes an improved version of RPCL. One measure is to adjust the value of α_r during the ith training epoch according to

$$\alpha_r = \left(1 - \frac{i-1}{e-1}\right) \times \alpha, \quad i = 1,2,\cdots,e, \quad (6)$$

where e represents the total number of training epochs; α represents the initial value for α_r. When k is set to 6 and 8 the clustering results using an improved RPCL algorithm are visualized in Fig. 6 and 7.

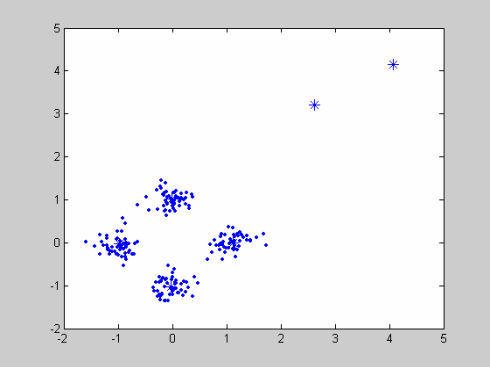

Fig. 6: The clustering result using an improved RPCL algorithm when k is set to 6. Note that there are 2 (k-4) redundant neurons that have been isolated from the cluster area and at the same time there is no cluster center that has deviated from the mean.

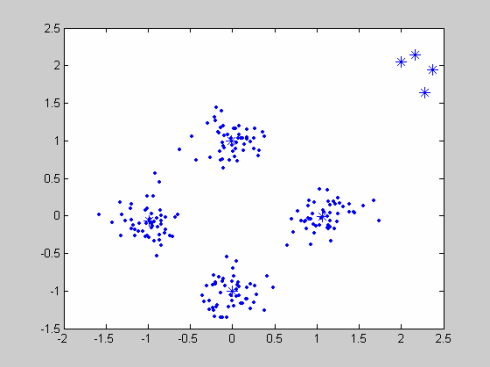

Fig. 7: The clustering result using an improved RPCL algorithm when k is set to 8. Note that there are 4 (k-4) redundant neurons that have been isolated from the cluster area and at the same time there is no cluster center that has deviated from the mean.

The improved RPCL observably enhances the clustering performance when k is given a comparatively large value at the beginning. It is able to expel those redundant neurons effectively from the cluster area while at the same time prevent the deviation of the cluster centers from cluster means. Therefore, we will adopt this improved RPCL as the competitive learning algorithm in the next section.

4. Experiments

We extract 2029 samples out of 508 stocks in Shanghai Stock Exchange which contain 593 continuation patterns and 1436 reversal patterns, to form the training set. We also extract 4937 samples, out of 155 stocks from Shenzhen Stock Exchange within the same time-span which contain 54 continuation patterns, 270 reversal patterns as well as 4613 neither of the two, to form the testing set. The training set is meant to train the classification network, while the testing set is meant to test the recognition ability of the classification network as well as to train the clustering network.

4.1. Clustering analysis based on SOM

In this section, the clustering result based on a two-dimensional SOM containing 81 (9×9) output neurons with 2000 training samples and 10000 training epochs is provided. After the training, SOM calculates the distance between the current sample and each output neuron, and categorizes the input sample into the cluster corresponding to the neuron whose output is 1. The weight vector of the neuron is also named as the prototype vector, corresponding to the clustering center. Since that SOM directly calculates the distance between the input vector and the weight vector of the neuron instead of calculating the inner product, it is not necessary to normalize the input vectors beforehand [17]. Fig. 8 visualizes the prototype vectors for 81 clusters while Fig. 9 and Fig. 10 are intended to show the two clusters with the most samples, whose numbers are 62 and 54, respectively. These two clusters represent the most frequently occurred patterns in stock curves.

As shown in Fig. 8, SOM clustering enables the resemblance among prototype vectors within the same neighborhood. For example, prototype vectors, with the position of (2, 7), (2, 8), (3, 7) and (3, 8) in Fig. 8, exhibit almost the same shape, which indicates that some clusters can be merged.

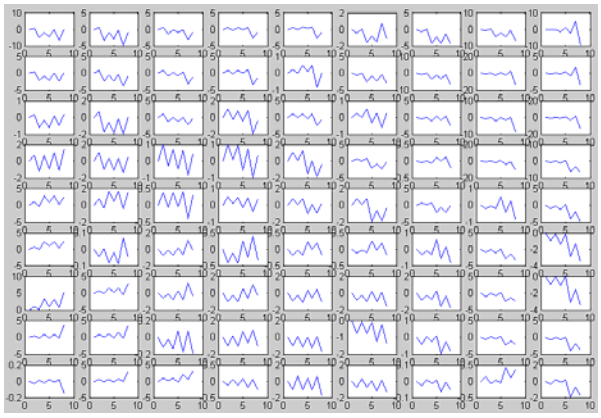

Fig. 8: Prototype vectors for SOM clustering analysis.

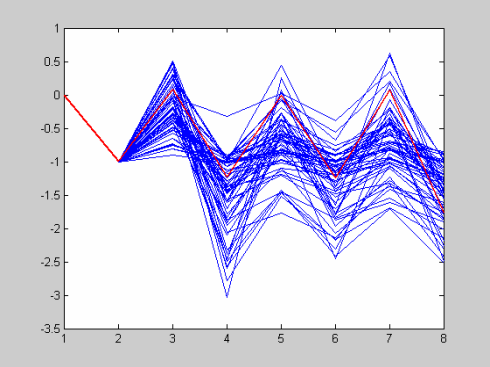

Fig. 9: Cluster 1 from the clustering result based on SOM, containing 62 samples out of 2000. The primary geometric feature of this cluster is that all local maxima or minima are approximately of the same value, leading to a volatility of stock price within two horizontal lines.

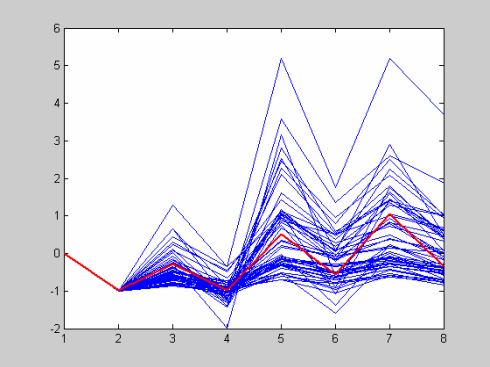

Fig. 10: Cluster 2 from the clustering result based on SOM, containing 54 samples out of 2000. The primary geometric feature of this cluster is that all local minima are approximately of the same value, while the local maxima are going through a gradual increment in value.

4.2. Clustering analysis based on RPCL

In this section, the clustering result based on a RPCL network containing 81 (9×9) output neurons with 2000 training samples and 200 training epochs is provided. After the training, RPCL calculates the distance between the current sample and each output neuron, and in the meantime categorizes the input sample into the pattern corresponding to the neuron whose output is 1. Fig. 11 visualizes the prototype vectors for the RPCL network, where it can be clearly observed that 21 of them, for example the (1, 7), (2, 9), (3, 3), (3, 9) and (5, 2), are considerably different from the others as they don't exhibit a seven-segment zigzag shape. As a matter of fact, such prototype vectors correspond to those neurons which have been isolated from the cluster area, indicating that there are only 60 (81-21) clusters within the sample space.

Fig. 12 and Fig. 13 visualize the two clusters containing the most samples, with 72 and 60 respectively. Similarly, they represent the two most frequently occurred patterns in stock curves.

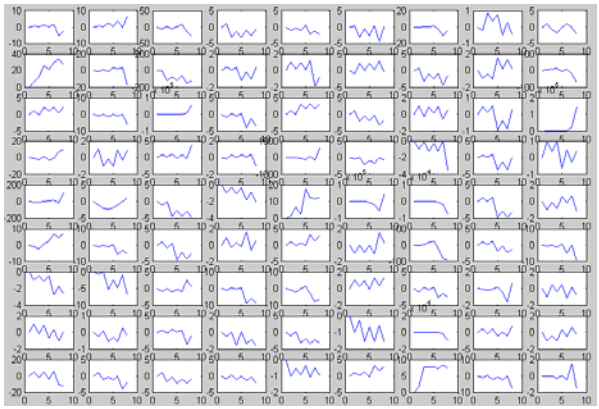

Fig. 11: Prototype vectors for RPCL clustering analysis.

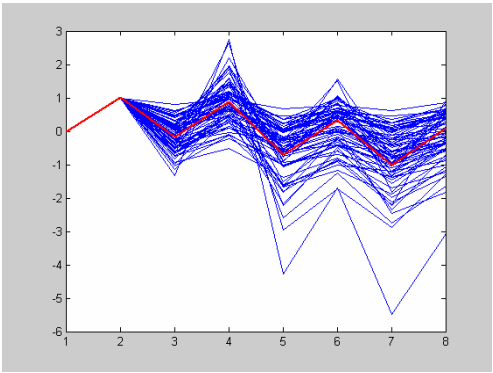

Fig. 12: Cluster 1 from the clustering result based on RPCL, containing 72 samples out of 2000. The primary geometric feature of this cluster is that both the local maxima and minima both go through a gradual decrement in value.

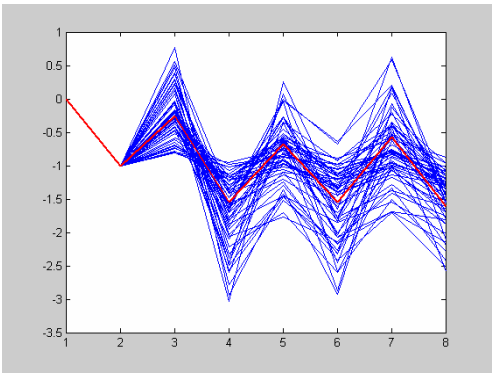

Fig. 13: Cluster 2 from the clustering result based on RPCL, containing 72 samples out of 2000. The primary geometric

feature of this cluster is that the first and third local maxima are approximately of the same value while the second one is comparatively smaller as well as that the second and third local minima are approximately of the same value, leading to that the third to the sixth segments constitute a "W" shape.

4.3. A comparison between the two results

First of all, we compare these two results according to the shapes of the prototype vectors. As shown in Fig. 8, each neuron resembles each other within its 1-neighborhood considerably, which indicates that some output neurons in SOM might be redundant. Therefore there are actually less than 81 clusters existing in the sample space. Nonetheless, RPCL automatically deletes the effects of those redundant neurons by isolating them away from the cluster area.

Second of all, we investigate into the distribution of samples within each cluster. Let $\vec{d}$ denote the number of samples of each cluster, with $\vec{d}(x)$ ($1 \le x \le 81$) representing the number of samples of the xth cluster. In addition, assume that H_i (i is a positive integer) denotes the number of clusters whose samples amount to at least (i-1)*10 and less than i*10. When SOM is utilized, it can be acquired that $0 \le \vec{d}(x) \le 62$ while the variance of $\vec{d}$ is 171.22. The number of clusters which have at least 30 samples is 25 and the total number of samples is 988. The value of H_i in SOM is shown in the second line of Table 1. Accordingly, when RPCL is utilized, it can be acquired that $0 \le \vec{d}(x) \le 72$ while the variance of $\vec{d}$ is 349.02. The number of clusters which have at least 30 samples is 37 and the total number of samples is 1532. The value of H_i in RPCL is shown in the third line of Table 1.

approach	H_1	H_2	H_3	H_4	H_5	H_6	H_7	H_8
SOM	10	18	28	17	2	5	1	0
RPCL	24	5	15	18	13	4	1	1

Table 1: Different values of H_i when i is of different values for both SOM and RPCL approaches, where H_i (i is a positive integer) denotes the number of clusters whose samples amount to at least (i-1)*10 and less than i*10.

A conclusion can be reached that a comparatively better equilibrium of the distribution of samples within clusters can be achieved when suing SOM approach, while the distribution is more concentrated when RPCL is adopted. Taking into account that the concentration exhibited in sample distribution helps substantiate that there does exist some frequently occurred patterns in stock time series, RPCL serves to be more preferable.

5. Conclusions

With the progressive complexity of stock markets, it becomes increasingly difficult to discover the underlying technical patterns. The implementation of automatic pattern recognition using computer technologies has served as an affective alternative. In this paper, an approach to effectively extract patterns out of stock time series is introduced along with clustering analyses conducted upon the patterns. With the aid of trend segmentation and pattern feature extraction, it becomes more convenient and effective to recognize patterns from stock time series. In the clustering analysis that follows, namely the SOM and competitive learning clustering analyses, experiments are carried out using as many as 2000 patterns gathered from 155 stocks in the Shenzhen Stock Exchange. The results demonstrate that RPCL is capable to achieve a better clustering result over SOM, substantiating that there does exist some patterns in real stock time series which tend to occur more frequently than the average.

Acknowledgement

This work is supported by National Nature Science Foundation of China (Grant No. 70571003).

References

[1] 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, 1995.

[2] W. Leigh, N. Paz and R. Purvis, Market timing: A test of a charting heuristic, *Economics Letters*, 77:55-63, 2002.

[3] 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*, 37:515-530, 2004.

[4] A. Lo, H. Mamaysky and J. Wang, Foundations of technical analysis: computational algorithms, statistical inference, and empirical implementation, *Journal of Finance*, 55:1705-1765, 2000.

[5] A. Sklarew, *Techniques of a Professional Commodity Chart Analyst*, Commodity Research Bureau, New York, 1980.

[6] S.C. Suh, D. Li and J. Gao, A novel chart pattern recognition approach: A case study on cup with handle, *Proc of Artificial Neural Network in Engineering Conf*, St. Louis, Missouri, 2004.

[7] W.J. O'Neil, *How to Make Money in Stocks*, 3rd Ed., McGraw-Hill Companies, New York, 2002.

[8] S. Anand, W.N. Chin and S.C. Khoo, Chart patterns on price history, *Proc of ACM SIGPLAN Int Conf on Functional Programming*, Florence, pp. 134-145, 2001.

[9] K. Kamijo and T. Tanigawa, Stock price pattern recognition: A recurrent neural network approach, *Proc of the Int Joint Conf on Neural networks*, pp. 215-221, 1990.

[10] T. Fu, F. Chung, V. Hg and R. Luk, Pattern discovery from stock time series using self-organization maps, *Workshop Notes of 7th ACM SIGKDD Int Conf on Knowledge Discovery and Data Mining*, San Francisco, pp. 27-37, 2001.

[11] L. Xu, A. Krzyzak and E. Oja, Rival penalized competitive learning for clustering analysis, RBF net, and curve detection, *IEEE Trans Neural Networks*, 4:636-649, 1993.

[12] L. Xu, How many clusters? : A YING-YANG machine based theory for a classical open problem in pattern recognition. *Proc of IEEE Int Conf on Neural Networks*, pp. 1546-1551, 1996.

[13] P. Guo, C. L. P. Chen, M. R. Lyu, Cluster number selection for a small set of samples using the Bayesian Ying-Yang model, *IEEE Trans Neural Networks*, 13:757-763, 2002.

[14] N. Bulkowski, *Encyclopedia of Chart Patterns*, 2nd Ed., John Wiley and Sons, 2005.

[15] F. Chung, T. Fu, V. Hg, R. Luk, An evolutionary approach to pattern-based time series segmentation, *IEEE Trans Evolutionary Computation*, 8:471-489, 2004.

[16] E. Keogh, S. Chu, D. Hart and M. Pazzani, An online algorithm for segmenting time series, *Proc of IEEE Int Conf on Data Mining*, pp. 289-296, 2001.

[17] M.T. Hagan, H.B. Demuth and M.H. Beale, *Neural Network Design*, PWS Publishing Company, Boston, 1995.