Classification of Extended Control Chart Patterns: A Neural Networks Approach

BUNTHIT WATANAPA AND JONATHAN H. CHAN

School of Information Technology
King Mongkut's University of Technology Thonburi (KMUTT)
91 Suksawasd 48, Bangmod, Thungkru, Bangkok, 10140
THAILAND

Abstract: - This paper generalizes the application of artificial neural network (ANN) by classifying six common control chart patterns into eight classes of time series data patterns. Incorporating two more patterns of bottom-out and peak-off can yield better insights for not only the traditional real time control environment but also the behavioral study in other time-domain systems such as money and security markets. This work reports the results of empirical study on the incorporation of new extracted features, especially those with a lesser extent of outliers' effect, for example median, robust regression and RMS value of the time series. The feedforward backprogation ANN is deployed and experimented using two different training schemes, namely the Levenberg-Marquardt method and the Bayesian regularization. The best performance generated by the ANN is 98% classification accuracy. Technical insights into the model settings are also provided.

Key-Words: - Time series data, Classification, Control chart patterns, Neural networks, Levenberg-Marquardt method, Bayesian regularization, Principal component analysis.

1 Introduction

The system of control chart is a well-known application of time-series data patterns to study the behavior of the underlying process in both real-time environment and off-line. Traditionally, there are six commonly classified patterns of interest which help to indicate the condition of the ongoing system: normal, cyclic, increasing trend, decreasing trend, upward shift and downward shift. Unfortunately, these patterns only help to confirm abnormalities after the events have occurred. For a shorter haul or in a real-time control system, the early detection of revival or reversal is more insightful for the tasks of monitoring and control. The confirmation of ending of uptrend or downtrend in stock price pattern is an example of such requirements. Thus, this paper proposes the use of two additional patterns of bottom-out and peak-off to supplement the six common control chart patterns.

Artificial neural network (ANN) is a prospective classifier of the above-mentioned eight classes due to its flexibility in modeling without prior knowledge of how classifier's determinants interact with each other. Also, ANN provides robustness on volume of input data via the model's parameter tuning process and architectural design. There are many proven records on the successes of ANN as being classifier for control chart pattern, such as the work of Lavangnananda and Piyatumrong in 2004 [1], Pham and Chan in 1998 [2], Sagiroglu *et.al*. in 2000 [3], and Guh *et al*. in 1999 [4]. A detail look at the use of ANN for pattern recognition with numerous other applications was given by Bishop back in 1995 [5] and Rapaso and Cruz in 2002 [6].

Comparing with traditional statistical techniques, ANN is more robustly effective especially when applying in an environment where data is noisy. The over-fit effect of the well defined mathematical model of statistics is overcome by the adaptability of ANN to new and noisy data via its learning ability.

However, the flexibility of ANN leads to a variety of architectural designs and learning approaches which possibly yield different performance level achieved on the same problem, as reported by Pham and Chan [2], Pham and Sagiroglu [7], and Sagiroglu *et al*. [3]. The difficulty lies in the design and tuning issues since there is no universal rule that guarantees best result of modeling. As presented in Section 4 later on, this paper also reports a satisfactory achievement of a

feed-foreward backpropagation network or simple MLP model and provides empirical proof on the importance of parameter settings.

Next section describes the empirical study on the simulated data representing all eight classes and the selection of attributes to classify them. Section 3 then develops the architecture of the ANN model with an illustration of the parameter tuning process. Results of simulation and issues of implementation are discussed in Section 4. Finally, the paper presents concluding remarks with a proposed future direction of work.

2 Input Data and Determinants of the Classifying Model

The section first describes the eight patterns of interests, especially how to generate the simulated data of the additional two classes. Then the process of selecting the classifier's attributes is provided.

2.1 Generating simulated time-series data

Focusing on the classification of time-series data obtained from the control chart system or alike, this study expands the scope of the traditional classifier by including two more generic classes: Peak-off and Bottom-out patterns. Figure 1 shows all possible patterns consisting of the two new and the other six standard patterns.

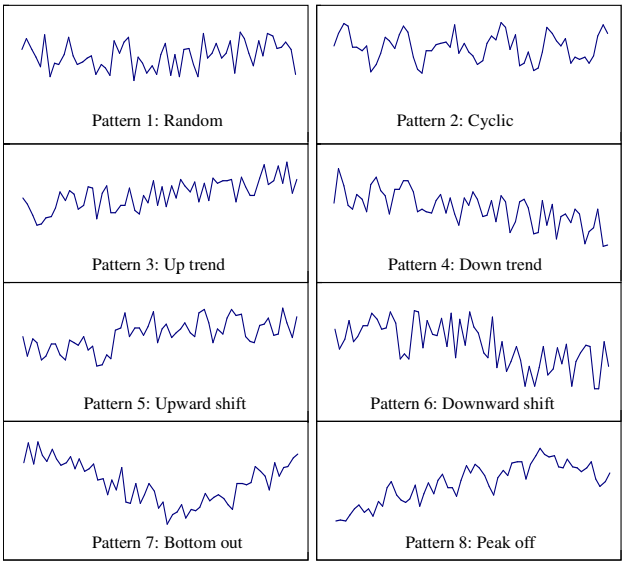

Fig. 1. Time series pattern to be classified

These two additional classes would make the model more applicable to other systems, e.g. mining for the selected patterns of the stock price movement time series data, detecting the sign of recovery or reversion of the on-line observation on any control patterns and systems.

Except for the additional two new classes, all data were synthetically generated with the standard schemes used in the existing literature such as those by Lavangnananda and Piyatumrong [1] and Sagiroglu *et al.* [3]:

$$\text{Randorm pattern: } y(t) = \mu + r(t)\sigma \quad (1)$$

$$\text{Cyclic pattern: } y(t) = \mu + r(t)\sigma + a\sin(2\pi t / T) \quad (2)$$

$$\text{Up trend pattern: } y(t) = \mu + r(t)\sigma + gt \quad (3)$$

$$\text{Down trend pattern: } y(t) = \mu + r(t)\sigma - gt \quad (4)$$

$$\text{Upward shift pattern: } y(t) = \mu + r(t)\sigma + ks \quad (5)$$

$$\text{Downward shift pattern: } y(t) = \mu + r(t)\sigma - ks \quad (6)$$

where $y(t)$ is the time series data at time t = 1, 2, 3, ..., 60, μ and σ are the mean and standard deviation of the simulated data , $r(t)$ is the random number generating function, a is the amplitude of the cyclic pattern, g stands for the slope of the series with upward or downward patterns, k is the point of the shifting, s is the step size for the shifting, and T is the period of the cyclic pattern.

Given Γ as the turning point of the trend, two new classes of peak-off and bottom-out patterns are generated in this work by using the following functions:

Peak-off pattern:

$$\begin{cases} y(t) = \mu + r(t)\sigma + gt \text{ for } t \le \Gamma \\ \quad = y(\Gamma) + r(t)\sigma - g \cdot (t - \Gamma) \text{ for } t > \Gamma \end{cases} \quad (7)$$

Bottom-out pattern:

$$\begin{cases} y(t) = \mu + r(t)\sigma - gt \text{ for } t \le \Gamma \\ \quad = y(\Gamma) + r(t)\sigma + g \cdot (t - \Gamma) \text{ for } t > \Gamma \end{cases} \quad (8)$$

2.2 Selection of classifier's parameters

Instead of raw time series data, Klösgen and Zytkow [8], Lavangnananda and Piyatumrong [1], and Nanopoulos *et al.* [9] reported success stories of extracting a limited set of dominating features and treating them as ANN's input. Adopting a feature-based neural network model, this research takes into account thirteen statistical measures representing characteristics of the 60-period time series data. These are listed below:

- Traditional statistical values: median (m), variance (σ^2), skewness, kurtosis;

- Correlation coefficients obtained from the robust regression using reweighted least square (slope, Y_0 or the intercept point on Y axis where t=0, and slope/SE where SE is the standard error measured for the estimation of the coefficient),
- The root-mean-square (RMS) of inputs which equals the norm or the Euclidean length (the square root of the sum square of the series of input which represent the magnitude of the data series) divided by the square root of the length of the series, or $\sqrt{\left(\sum_{t=1}^{n} x^2(t)\right)/n}$.
- The approximation of the integral of the series using trapezoidal numerical integration with a unit spacing.

The Median and the three estimated coefficients of correlations obtained from the robust regression scheme are selected because of their smoothing function which aims to limit the effect of outliers. The robust regression applies an iteratively reweighted least squares algorithm. Starting from the ordinary least square in the first round, the reweighing process is done iteratively basing on the bi-square function of the residuals from the previous iteration:

$$weight_t = \left[\left| \frac{r_t}{tune * s * \sqrt{1-h}} \right| * (1 - r_t^2) \right]^2 \qquad (9)$$

where r_t is the residual of data at time t from the previous iteration, s is an estimate of the standard deviation of the error term. In this context, s = MAD/0.6745, where MAD is the median absolute deviation of the residuals from their median and the constant 0.6745 makes the estimate unbiased for the normal distribution. The other two variables are the vector of leverage values from a least squares fit (h) that adjust the residuals by down-weighing high-leverage data points with a large effect on the least squares fit and the tuning constant of 4.685 (*tune*) which helps to standardize the adjusted residuals. Overall, this algorithm gives lower weight to points that do not fit well. For more detail, reader is referred to the work of DuMouchel *et al.* [10] and Street *et al*. [11].

The other three traditional statistical measures have been proven as valid extracted features for classifying into six classes based on a synergistic ANN [1]. The incorporation of the remaining two measures of RMS and integral of the series are to complement the system with spatial features.

To enhance the ability in detecting the reversion of trend data, the 60-periods of time series data is chopped into 2 partially-overlapped windows each with 45 periods as shown in Figure 2. All those four traditional statistical features are extracted from each window and then the resulting subtractions of each pair of the corresponding features are collected.

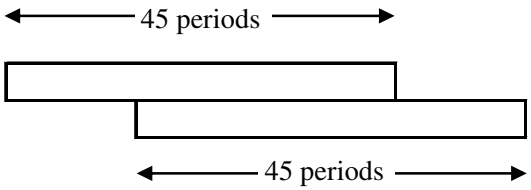

Fig. 2. Division of time series data

Thus, there are another four features representing the difference of features in the two windows signified as 1 and 2, respectively:

m_2-m_1, $\sigma_2^2 - \sigma_1^2$, skew$_2$-skew$_1$ and kurtosis$_2$-kurtosis$_1$.

3 Neural Network as the Classifier

The multilayer feed-forward back-propagation neural net was selected in this research due to its speed and satisfactory performance. In the next subsection, we discuss input preprocessing and selection of the non-redundant set of inputs based on principle component analysis (PCA), follow by the structure of ANN with design of experiment.

3.1 Non-redundant set of determinants and pre-processing

To optimize the speed and performance of the ANN, all 13 inputs were pre-processed so as to standardize their values to have a zero mean with standard deviation of one. This is followed by analysis with the Principal Components Analysis (PCA) method [12] so that only those input factors with a certain confidence level of being independent from the others would be chosen as the representative set of the classifier's attributes.

In brief, using PCA, the covariance is generated from all 13 extracted features, then the eigenvectors and eigenvalues are calculated from the resulting square matrix of covariance. Eigenvalue represents the magnitude of contribution to the total variance in the data set. Those features (eigenvectors) with an eigenvalue of less than the specified value, e.g. 2% of total variance for this context, will be excluded.

The experiment was conducted on three sets of data with a total number of 800 series of 60-period

data, separating into 70 series of training data for each class, 15 series of validating data for each class, and 15 series of testing data for each class. The validating set of data is to make sure that the resulting model is not overfitted, which may happen due to the limited number of training data. All data were generated with a high level of noise so that they reflect difficult situations in real-time control systems, especially the simulated data of the existing six patterns that have been adopted from the work of Lavangnananda and Piyatumrong [1].

3.2 ANN model and design of experiment

Empirically, according to the result of PCA test with a threshold of 2% contribution to to total variance, only nine of those thirteen extracted features are indicated as uncorrelated inputs. To gain technical insights, the feed-forward backpropagation ANN is trained in two different schemes, which are Levenberg-Marquardt (trainLM) [13] and Bayesian regularization (trainBR) [14]. In brief, trainLM manipulates the gradient descent with small step size at the start and swaps quickly to the Newton-alike method, which is faster and more accurate when reaching near an error minimum. According to Hagan and Menhaj [13], trainLM can train the ANN with a much faster rate than the traditional gradient descent. The latter training scheme, trianBR, may be viewed as a smoothened version of Levenberg-Marquardt. It optimizes the generalization quality of network training by minimizing a linear combination of squared errors and weights. By the virtue of its generalization quality, it should be less prone to overfitting the input data, given the same architecture or set of inputs.

Based on our empirical tests, the best structure and the parameters set for each scheme are different. Using trainLM, the MLP performs the best when organized in a 9-8-8 fashion which represents the number of nodes in the input layer, single hidden layer, and the output layer, respectively. With trainBR, the best organization is 9-9-9-8 where 9 nodes are required in 2 hidden layers. Both models require a common set of training parameters as summarized below:

Transfer function:
Hyperbolic tangent sigmoid (Tansig)
Maximum epoch: 100;
Performance goal: 0;
Maximum validation failure: 5;
Minimum performance gradient: 1e-10;
Weight/bias learning function:
Gradient descent with momentum weight and bias

Even though trainBR requires the performance function of SSE (sum square error), the experiment shows that trainLM performs better with function of MSE (mean square error). Accordingly, we set the different performance functions on each method.

The model was trained, validated and tested in 10 separate runs with 3 replicates each. Data from each replicate is randomly re-sampled from the original set of 800 time-series data. The resulting performance is reported in the next Section.

4 Results of Experiment and Discussion

This section presents the results of experiment and gives some insights into the technical issues. From this point onwards, we refer the model with trainLM as model A and the model with trainBR as model B for brevity.

4.1 Experimental results

The results of experiment are summarized in Table 1 below. For both Models A and B, the accuracy of the model is around 95% on average, with the best result of 98.3% where only 2 falses out of 120 classifications were found.

Table 1. Experimental results for 10 replicates

Training approach	Average Accuracy	Best Accu.	Elapse time (s)	Epochs
Model A	95.0%	98.3%	35	32-45
Model B	95.3%	98.3%	40	22-40

Even though Model B seems to take a bit longer than Model A computationally, there is no significant difference. More details of experimental results for each replicate is provided in Table 2.

Table 2. Results of experiment in each replicate

Running number	Classification accuracy	
	Model A	Model B
1	0.93	0.95
2	0.95	0.95
3	0.94	0.97
4	0.98	0.93
5	0.98	0.98
6	0.94	0.93
7	0.94	0.94
8	0.94	0.96
9	0.96	0.97
10	0.94	0.95
Average accuracy	0.950	0.953

As seen in Table 1, in most cases, the trainings were forced to stop due to the failure of validation vector exceeding the maximum value of 5 epochs to improve the network performance. Figure 3 shows the patterns of MSE improvement over the training session with a stagnancy when validation approaches epoch 28 that forced the training process to stop on epoch 43. Empirically, the result with no validation gave a very much lower MSE but worse accuracy (less than 90%). The validation is proven in this context as a useful tool to prevent ANN from overfitting and greatly saving computational time.

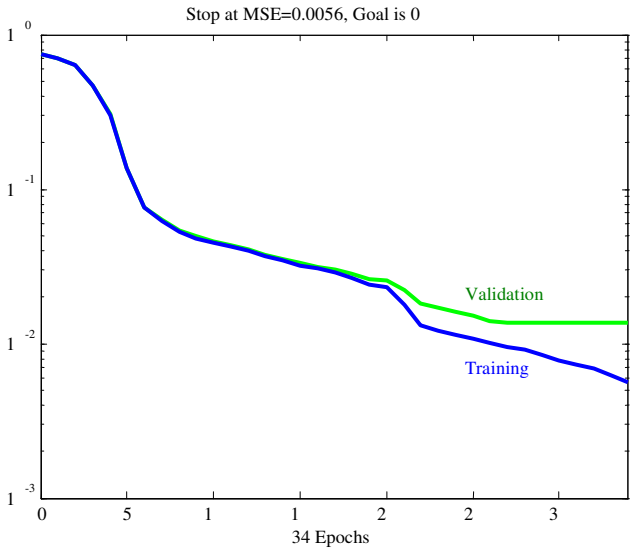

Fig. 3. Converging MSE improvement during training with a stop due to stagnant improvement on validation.

Empirically, Model B is less sensitive (in terms of accuracy) to the change in model organization. For example, Model B could give around 94% accuracy when organized in a 9-8-8 model but Model A gave adverse results (less than 90% accuracy) when organized in a 9-8-8-8 model. Model B required a longer lapse time but fewer number of training epochs.

In conclusion, the experiment has proven that ANN is an effective model for classifying all eight control chart patterns of requirement. Model B (Bayesian regularization training) is found to have the same performance as Model A (Levenberg-Marquardt) but with a higher robustness or generalization quality. That is, the Bayesian regularization training algorithm is less sensitive to the various organizations of ANN models, although the learning algorithm of Levenberg-Marquardt is a bit faster.

4.2 Discussion and insights

There are a couple of additional insights to discuss according to the experimental results. First of all, the simple and popular multilayer feed-forward backpropagation network works very well in this context. The relevant factors are the quality of attributes, the selection process of PCA and the normalization of input scales. The justification for the first two factors is that when the attributes or determinants of the classifier is closely correlated with the pattern, then it would require less complex model to capture the behavior of the pattern of interest. The justification for the latter factor is that the equalized scale avoids bias and gives a flat level of importance to every determinant in determining the outcome.

Secondly, the achievement obtained from this simple model on distinguishing the eight control chart patterns, involving the extra two peak-off and bottom-out ones, indicates an opportunity to apply this classification method out of the traditional control chart pattern applications. For example, this methodology may be incorporated as an integral part of the forecasting or mining for stock price pattern or other trendy time-series data.

5 Conclusions and Future Direction

A number of extracted features with smoothing effect, e.g. correlation coefficients from the robust regression, median, and difference of statistical measures from two overlapped windows, have been combined into the determinants of the classifying model that differentiates control chart patterns. In addition to the six standard patterns, two more patterns, i.e. peak-off and bottom-out, are added so that the classifier is generalized and can be applied to wider applications, e.g. stock price patterns. A simple feed-forward backpropagation neural network is modeled with two distinct training methods: Levenberg-Marquardt and Bayesian regularization. From simulated experiments, some insights are given. (i) A simple MLP network is proven to be an effective model for classifying the eight patterns of time-series data, with the aid of appropriate features and preprocessing. (ii) Different learning approaches can lead to different effective architecture of ANN (e.g. number of hidden layers).

Potential directions for future work are, for example, to use a more sophisticated ANN model, e.g. synergistic organization, self-organizing structure, *etc.*, so that the overall performance can be improved. Another interesting direction is to employ the model into a system with real time-series data with significant economic values, e.g.

incorporating it as an integral part of the forecasting system or decision support system for buy-sell-hold operations in a specific stock market.

6 Acknowledgement

The authors wish to express their gratitude to Dr. Kittichai Lavangnananda for his advice and supports.

References:

[1] K. Lavangnananda and A. Piyatumrong, "Utilizing Features Extraction in Classifying Control Chart Signals : A Synergistic Neural Networks Approach," *2004 International Conference on Advances in Intelligent Systems - Theory and Applications (AISTA2004)*, November 15-18, Centre de Recherche Public Henri Tudor, Luxembourg-Kirchberg, Luxembourg, 2004.

[2] D.T. Pham and A.B. Chan, "Control Chart Pattern Recognition using a New Type of Self-Organising Network," *Proc. Instn. Mech. Engs.*, Vol. 212, No. 1, 1998, pp. 115-127.

[3] S. Sagiroglu, E. Besdok and M. Erler, "Control Chart Pattern Recognition Using Artificial Neural Networks," *Turk J Elec Engin*, Vol.8, No.2, 2000, pp. 137-147.

[4] R.-S. Guh, Zorriassatine, F., Tannock, J.D.T. and O'Brien, C.O., "Online Control Chart Pattern Detection and Discrimination – A Neural Networks Approach", *Artificial Intelligence in Engineering* vol. 13, 1999, pp. 413-425.

[5] C.M. Bishop, *Neural Networks for Pattern Recognition*, Oxford: Clarendon Press, 1995.

[6] R.C. Rapaso and A.J. Cruz, "Stock Market Prediction Based on Fundamentalist Analysis with Fuzzy-Neural Networks," *Proc. of the 3rd WSEAS Int. Conf. on Neural Networks and Applications (NNA'02)*, Interlaken, Switzerland, 12-14 February 2002, pp. 58-62.

[7] D.T. Pham and S. Sagiroglu, "A Comparative Study of Four Multilayered Perceptron Training Algorithms," *Int. Journal of Machine Tools and Manufacture*, Vol. 41, 2001, pp. 419-430.

[8] Klösgen, W. and Zytkow, J.M., *Handbook of Data Mining and Knowledge Discovery*, Oxford University Press, 2002, pp. 214-218.

[9] A. Nanopoulos, R. Alcock and Y. Manolopoulos, "Feature-Based Classification of Time-Series Data," in *Information Processing and Technology*, N. Mastorakis and S.D. Nikolopoulos, Eds., Nova Science Publishers, Cormack, NY, 2001, pp. 49-61.

[10] W. DuMouchel and F. O'Brien, "Integrating a Robust Option into a Multiple Regression Computing Environment," in *Computing Science and Statistics: Proceedings of the 21st Symposium on the Interface*, (K. Berk and L. Malone, eds.), American Statistical Association, Alexandria, VA, 1989, pp. 297-301.

[11] J.O. Street, R. J. Carroll, and D. Ruppert, "A Note on Computing Robust Regression Estimates via Iteratively Reweighted Least Squares," *The American Statistician*, vol. 42, 1988, pp. 152-154.

[12] I.T. Jolliffe, *Principal Component Analysis*, New York: Springer-Verlag, 1986.

[13] M.T. Hagan and M. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Transactions on Neural Networks*, vol. 5, no. 6, 1994, pp. 989-993.

[14] F.D. Foresee and M. T. Hagan, "Gauss-Newton approximation to Bayesian regularization," *Proceedings of the 1997 International Joint Conference on Neural Networks (IJCNN'97)*, Houston, Texas, USA, 1997, pp. 1930-1935.