Proceedings of the International Multiconference on
Computer Science and Information Technology, pp. 195 – 200
ISBN 978-83-60810-14-9
ISSN 1896-7094

Discovery of Technical Analysis Patterns

Urszula Markowska-Kaczmar
Wrocław University of Technology,
Wybrzeże Wyspiańskiego 27,
50-370 Wrocław, Poland
Email: Urszula.Markowska-Kaczmar@pwr.wroc.pl

Maciej Dziedzic
Wrocław University of Technology,
Wybrzeże Wyspiańskiego 27,
50-370 Wrocław, Poland
Email: 133644@student.pwr.wroc.pl

***Abstract*—In this paper our method of discovering data sequences in the time series is presented. Two major approaches to this topic are considered. The first one, when we need to judge whether a given series is similar to any of the known patterns and the second one when there is a necessity to find how many times within long series a defined pattern occurs. In both cases the main problem is to recognize pattern occurrence(s), but the distinction is essential because of the time frame within which identification process is carried on. The proposed method is based on the usage of multilayered feed-forward neural network. Effectiveness of the method is tested in the domain of financial analysis but its adaptation to almost any kind of sequences data can be done easily.**

I. Introduction

THE issue of discovering data sequences has been heavily investigated by the scientists of different disciplines for many years. Despite this fact there is no doubt the issue is still up-to-date. Statisticians, economists, weather forecasters, operating system administrators – all of them, in their daily routine, deal with many kind of sequences. Specifically, in the domain of finance analysis there are patterns defined by the *Technical Analysis* (TA). Recognition of some of this patterns among quotation data triggers investors buy or sell decisions regarding examined stock. So it is crucial for the people who play the stock exchange to recognize patterns when they are really formed by stock exchange quotations. Because of that there is a need to provide trustworthy method of finding defined sequences. Lately, discovery of patterns in time series plays very important role in the area of bioinformatics [2] also.

In this paper a method of discovering data sequences in the domain of financial analysis is presented but its adaptation to any other kind of sequences data can be easily done. This method uses multilayered feed-forward neural network to recognize the technical analysis patterns. All experiments which aim was evaluation of the method efficiency, are done by the use of data which come from the Warsaw Stock Market.

The paper consists of five sections. The next one describes different approaches to the problem of sequence data discovery. Our method is introduced in the third section. The next one presents the results of the experiments. some of them were performed by the use of the method in artificial environment simulating the Warsaw Stock Market. The final section presents conclusion and future plans.

II. Related Works

Methods of pattern discovery in time series sequences in the financial analysis are closely connected to econometrics which can shortly be defined as the branch of economy that deals with defining models of different systems by the use of mathematics and statistics. Some of these models are created by economists in order to make analysis of data or to make a prediction of future stock exchange quotations. The problem is to prepare a good model, where 'good' means the model which takes into consideration all important relations which can be distinguished in the modeled reality. This is of course not easy. Often some relations become important under certain circumstances when others turn out to be useless. To comply with all defined requirements there is a need to prepare accurate model which can consist of even hundreds of equations. Such approach causes difficulties in its comprehensibility by the user but also in a computer implementation. That is why scientists look for other methods of discovering patterns in time series.

Fu and others [3] describe a method which uses *perceptually important points* (PIPs) of the graph to compare it with other graph. By PIPs are assumed points that are significant for the shape of the diagram to which they belong. Authors presented a method for finding PIPs and algorithms for determining the distance between points from two different graphs. The idea introduced by them reflects the human-like way of thinking (people usually do not remember all the points which build the graph – they keep just more significant ones in mind and then compare them to the other important points). The advantage of this algorithm is its easy implementation. Despite of that fact, there is a big disadvantage of this method of discovering sequences. A problem is with series which have high amplitude between two adjacent points – higher than some PIPs can place between those two points. It leads to the problem, when we have PIPs identified not among whole series but mainly in some its parts. Similar approach is applied in the paper [1], where a special metrics of similarity between a pattern in question and a given pattern is designed.

The usage of rules and fuzzy rules in searching time sequence patterns are considered as well. The examples can be found in [7] and [3].

Many researches are made by the use of machine learning methods in order to retrieve some predefined technical analysis patterns within the time series, e.g. [5].

Very popular approach is an application of Kohonen's neural network to cluster patterns retrieved among stock exchange quotations. The examples of SOM networks can be found in [4] and [6]. Authors admitted that this kind of network in their experiments showed good results in searching for patterns of main trend of quotations. They also consider this approach as not ideal for making predictions of turning points among quotations.

Other approach which used neural networks is presented in [5]. The method described in this paper can be shortly characterized as follows. Each of the patterns is memorized as a chart in the computers memory within some specified boundaries. Next, neural network (NN) is trained of chosen pattern. After training, the network is able to recognize whether a given series is similar to the pattern it was trained. To become results more trustworthy the author suggested to use two different NNs for a recognition of one pattern (the average of both results was treated as a final result). What is important, both neural networks had to be trained using different sets of learning patterns. The method based on chart pattern recognition in time sequence is proposed in as well.

III. The Details of the Method

Our method of discovering data sequences in time series is also based on the neural network which has feedback connections. It is trained with back propagation learning algorithm. The whole idea is simple. For each pattern of technical analysis one dedicated neural network exists which is trained to recognize it. The architecture of the network used in the experiments is presented in Fig 1. The network is fully connected. Each of the inputs represents exactly one value of a stock exchange quotation. In this figure *N* describes the number of input neurons (which was set to 27 in the experiments), *L* represents the number of hidden neurons (it was equal to 14 in the experiments) and *M* is the number of output neurons (that was set to 1). The response of the output neuron indicates whether a given series is recognized as a pattern that the network was trained to recognize.

To be more precise it is worth mentioning, that sigmoidal function was used as an activation function. This means that the value returned by the output neuron is in the range (0; 1).

The output value closer to the upper bound of the range was interpreted as a given series was similar to the series from training set. When the continuous range of values is allowed the obvious question is how to make a binary decision if the series represents a pattern in question or not? The answer is not so well-defined. It depends on what the parameters of the network training were set, what the stop criteria of learning algorithm were adjusted or what kind of activation function was chosen. In the experiments after preliminary experiments this threshold value was set to 0.85. In the Fig. 2 there are presented main steps of the proposed method of discovering data sequences. In the first step training patterns for neural network are prepared. It is important to provide representative patterns. It is a good practice that some of them should be multiplied within the training set (with added noise).

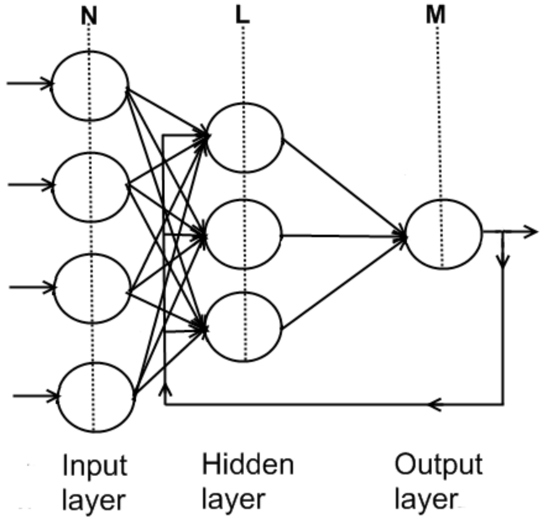

Fig. 1. The neural network architecture used in the experiments

```
PrepareTrainingPatterns()//define training set
NormalizePatterns()//prepare normalization
TrainNeuralNetwork()//training process
SmoothInputSeries()//step is optional
NormalizeInputSeries()//series' normalization
ProceedtheSeries()//Classifying decision
```

Fig. 2. The algorithm of discovering technical analysis patterns in time series

Adding similar learning patterns ensures that the neural network after the training process will have better generalization skills. The next step (normalization of patterns) is needed in order to reduce all defined patterns to a common range. It is important, because in other case series defined on different ranges could favor some patterns with higher values. Each value *s* from a series *S* is normalized according to the equation (1). In the next step the neural network is trained. The training process should be continued until an output value of the network reaches satisfied value (usually below defined threshold).

$$s_{norm,i}=\frac{s_i-\min(S)}{\max(S)-\min(S)}, \quad (1)$$

where: $s_i \in S$, $\min(S)$ – minimum, $\max(S)$ – maximum.

In the next step a given series, in order to be processed by the neural network, can be smoothed. It is especially essential when a series consists of any abnormal values. The aim of smoothing is to reduce the number of points where the amplitude between two adjacent points in the chart is extremely high. An example of smoothing result is presented in Fig. 3. Because this method changes the original points in a chart it is recommended to use it only if needed.

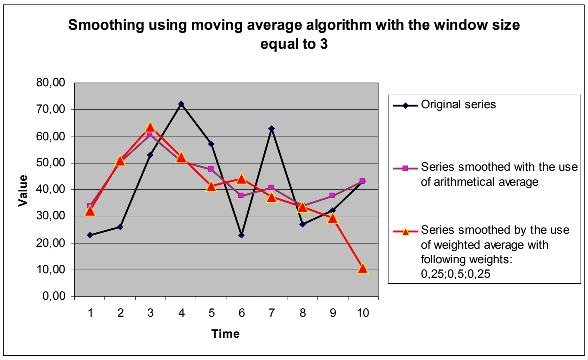

Fig. 3. An example of smoothed series

Because of the fact that patterns used during a training process were normalized, similar action has to be done after an optional smoothing of the series. It is crucial to have series defined on the same range as training patterns are. Otherwise the result cannot be reliable. The final step relies on processing values from the series given to the input layer of the network and calculating its output.

The algorithm shown in Fig. 2 can be easily used when the series (P) in question is of the same length as the patterns from the training set (series S).

The problem raises when the lengths are different. Having patterns longer than a series in a training set leads to the necessity of expanding a given series (i.e. by adding some additional points between existing ones). Depending on the shape of series which needs to be stretched, different methods should be used. A simple approach can be realized by the use of linear function to calculate values of extra points while more complex can demand the usage of more sophisticated curves in order to determine points values (such as Bezier curves).

The other case that should be considered is when a given series P is longer than the number of inputs in the neural network (a length of training patterns). Then a shortening of examined series should be done. In this case to solve this problem the following solutions can be suggested:

a) Shortening by a deletion of surplus points,
b) Shortening by a determining only perceptually important points (based on the idea presented in [3]),
c) Shortening by compressing a series.

The first technique is based on the assumption that some points from a time series can be removed without affecting its shape too much, which is especially true if concerned are series taken from the real stock exchange. In this case each subseries formed as hop or valley on the chart consists of many points which values change gradually. Removing one point from such short subseries will not affect a whole form. The simplest way to determine which points should be removed is to count how many of them is surplus (s_p). Afterwards the number of all points (m) in the series should be divided by s_p resulting in the steps (k) which should be used while designating indexes of surplus points within a given series. Fig. 5 presents the effect of the usage of the mentioned method to the series which is depicted in the Fig. 4.

The second technique is to find within a series exactly n characteristic points (called perceptually important points -*PIP*). Other points, which were not considered as characteristic points should be removed.

The last technique of shortening series of length m to become one with n values is its compressing. The compression can be done by specifying n segments in a given series and all values within each segment are substituted by one value. This value is an arithmetic or a weighted average of the substituted values (this method is a little similar to smoothing). The exemplary results are shown in Fig. 6.

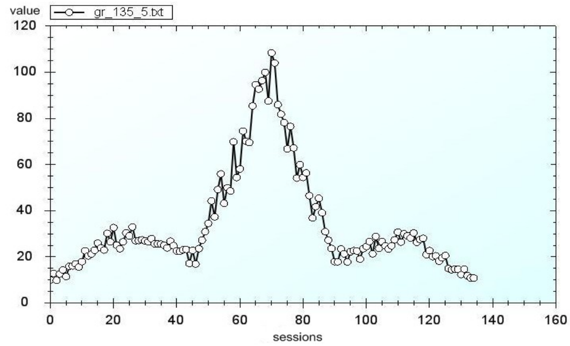

Fig. 4. The chart of 'Head and shoulder' pattern made of 135 points

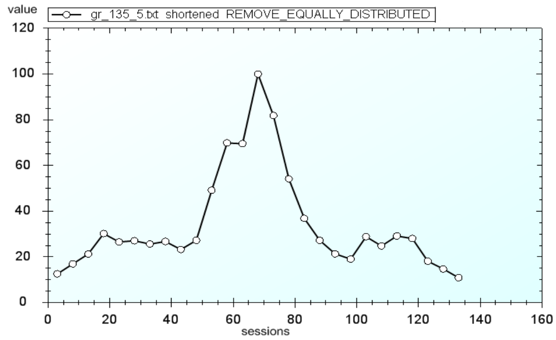

Fig. 5. The chart of 'Head and shoulder' pattern shortened to 27 points using a deletion of surplus points

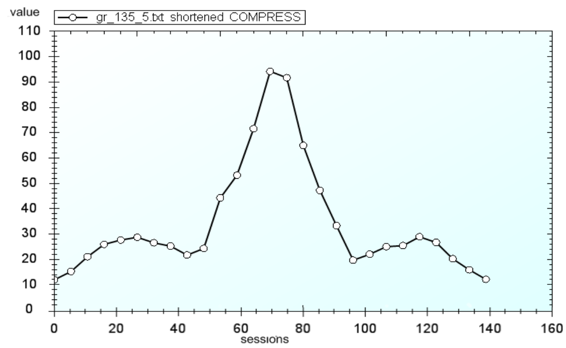

Fig. 6. The chart 'Head and shoulder' pattern shortened to 27 points using a compression

It is important to emphasize that all described previously issues were adequate to recognize a whole series as a pattern. The other case is when we want to find how many times an interesting pattern was repeated among the whole series (this operation can only be done, when a series is longer then training patterns). One approach to this problem is to specify start index and the number which represents a value of length step (which will be used for moving the window from the start index). Next, moving a window (which has the length equal to the number of input neurons of the neural network) the main series can be cut into subseries with a defined step from the start index. Then, each subseries should be checked whether it is similar to the pattern trained by the network. The problem becomes more complex when the length of subseries differs from the length of training patterns. We can consider checking the subseries of length from 2 up to *m* (where *m* represents the length of whole series).

In this case the problem occurs that computational complexity becomes $O(m^2)$. To reduce the number of subseries that should be checked, similarly to [3] a function *TC* (given by eq. (2)) is used. Its task is to control a length of a series which should be processed. This function returns a smaller value when the length of the series is closer to the preferred length. In eq. (2) *dlen* is the desired length of series (which in our case should be equal to the number of input neurons in the network), *slen* means the series length. Additionally, *dlc* parameter can be adjusted according to the steepy of the function which is used. Only for the points which are below specified threshold (i.e. $\lambda=0.2$) on the *TC* function graph the checking should be performed.

$$TC(slen, dlen)=1-\exp^{-(d_1/\theta_1)^2}, \qquad (2)$$

where $d_1=slen-dlen$, $\theta_1=dlen/dlc$

Fig. 7 illustrates how on the basis of *TC* function the lengths of subseries are checked. The following values of parameters were used: $dlen=180$, $dlc=2$, as *slen* all values of series were provided. For assumed value $\lambda=0.2$ red bolded line marks the range of the lengths to be checked.

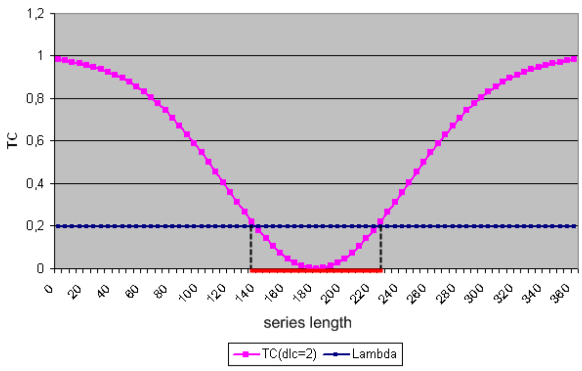

Fig. 7. Example of *TC* function usage

IV. Experimental Results

In the first experiment the shortening methods of series were evaluated in order to choose the best one. The network was trained to recognize the technical analysis pattern ‘head and shoulder’. The training set was prepared where each training pattern had the length 27 (the number of neural network inputs). It consists of positive training patterns (representing ‘head and shoulder’ form) as well as negative ones (that do not represent this form). The network was trained with an error equal to 0.001.

To evaluate the method of a shortening series the testing set was created. It contained: 30 artificial series of ‘head and shoulder’ pattern with the length equal to 54; 81 and 135 (10 of each length), 10 series of ‘triple top’, ‘double top’ and some randomly chosen patterns, finally some series of archive stock (GPW) exchange quotations (which were manually annotated by the authors whether they represent the pattern in question - ‘head and shoulder’ form or not. In these annotations the value 1 informs that a given time series represents a given pattern, value 0 means that it does not).

In the test it was arbitrary assumed that the network output equal or greater than 0.9 represents the neural network recognition of the pattern in question.

For each pattern from the testing set an *error* between desired value of the output and the one returned by the network was used to evaluate the results (absolute value of subtraction of mentioned elements). The average error calculated for each method is the basis of comparison. The results are shown in Table I.

Table I
Comparison of shortening techniques

Shortening technique	Average error	Deviation of average error
Surplus points	0.0672	0.1233
Compression	0.0697	0.1409
PIP	0.0936	0.1632

It is easy to notice that the best results are achieved by the method *surplus points*. The results in Table II show that it performs discoveries of patterns in the best way, as well. Effectiveness of patterns discovering was calculated as a relative number of properly recognized patterns to the number of all patterns.

Table II.
Effectiveness of discovering patterns using different shortening techniques

Shortening technique	Effectiveness		
	artificial series	GPW series	All series
Surplus points	1.0000	0.8730	0.9624
Compression	1.0000	0.8571	0.9577
PIP	0.8400	0.8413	0.8404

Based of an analysis of the results we can draw the conclusion that the proposed method of checking whether a given long series (longer than the number of inputs in the network) is similar to a chosen pattern returns very good outcome. For all presented techniques of shortening we can observe that effectiveness is greater than 80%, considering two best techniques we received even better result (~95% of properly classified series).

The aim of the next experiment was to check whether the methods of discovering patterns are sensitive to the length of tested subseries. For the test purpose one long series was chosen. It was created on the basis of stock exchange quotations of the stock market 01NFI from 150 sessions (from 14 August 2006 till 16 March 2007). The algorithm of discovering patterns has run twice. In the first run the length of the window varied from 2 to 100. In the second one the *TC* function was applied. It allowed to limit the number of time sequence lengths to be checked (the range from 22 to 32 for the network with 27 input neurons). The results are presented in [8]. The blue line represents the widths of window for which patterns could be found in the given series without using *TC* function, while the pink line shows the number of discovered patterns with the use of *TC* function. It can be easily notice that its usage really limits the range of widths to the (21; 32). The fact, that for the widths of window in the range 60 – 69 so high number of patterns were found can be a surprise. But it is nothing extraordinary. We have to keep in mind, that the neural network with well performed preprocessing algorithm (which properly shortens or expands series) can effectively recognize patterns regardless of the length of checked series. The obtained results show that the method is not very sensitive to the length of the tested time series.

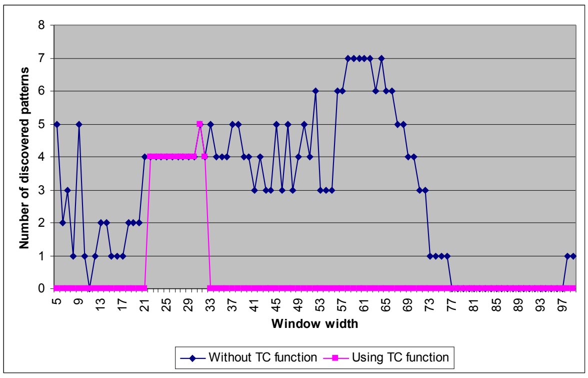

Fig. 8. The number of discovered patterns in relation to the window width

In Fig. 9 and Fig 10 examples of the series found during the experiment are presented (the red line represents a shape of the chart 'head and shoulder').

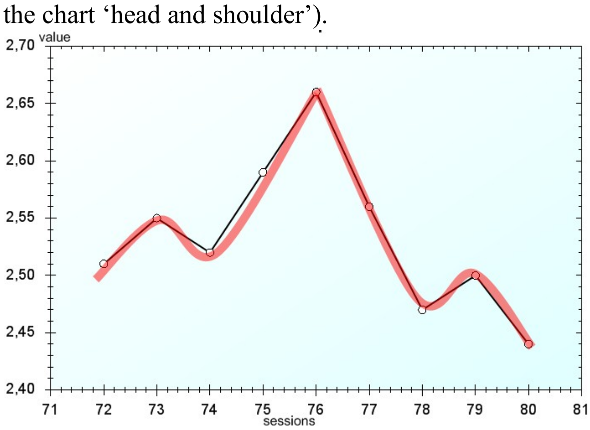

Fig. 9. Stock exchange quotations of 01NFI formed from 9 sessions identified as a 'head and shoulder' pattern

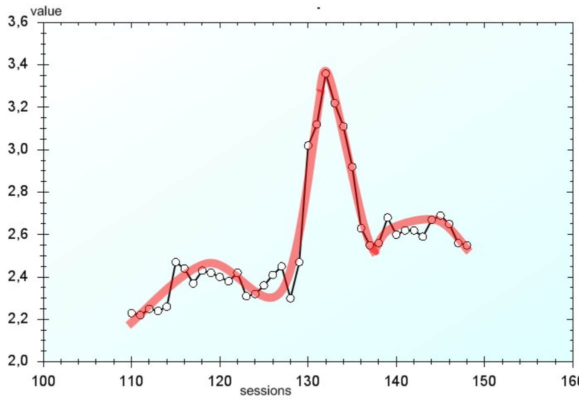

Fig 10. Stock exchange quotations of 01NFI formed from 39 sessions identified as a 'head and shoulder' pattern

As it was mentioned before, the method of discovering data sequences in a time series was tested also in the artificial environment – multi-agent stock exchange system which is presented in . In this system agents representing real investors are evolved by genetic algorithm. Each agent is described by the set of its coefficients defining its behavior. The aim of the system is to find the set of agents (with the best suited values of coefficients) who will be able to generate the stock price movement similar to the existing one in the real stock. Evolution takes place in steps which are called generations. After each generation the individuals (the set of agents in our case) are evaluated in terms of fitness value that informs about a quality of an individual. The better is the fitness value, the better is the set of agents (individual). Originally the system had an naïve algorithm (assigned as *old*) of identification which investments should be done by an agent. Then this algorithm was substituted by the method of discovering time series sequences presented in this paper (called *new*). The comparison of the results with the usage of both methods is shown in Table III.

An analysis of the results in the table clearly shows that the application of the new method improves the value of agents' fitness. The old algorithm returned good results only in the third test. It means that stock prices generated by the agents using newer decision algorithm are much more similar to the real ones. However, because a genetic algorithm has embedded randomness in its nature, more tests are required to fully evaluate the results which were not possible to perform now because of the duration of one experiment.

TABLE III.
THE COMPARISON OF NEW AND OLD ALGORITHM OF TAKING DECISION BY THE AGENTS

Nr	**Average fitness of all individuals in all generations**		**Fitness of the best individual in the experiment**	
	Decision algorithm			
	old	*new*	*old*	*new*
1	0.12	0.25	0.47	0.51
2	-0.01	0.19	0.43	0.45
3	0.11	0.05	0.59	0.50
4	0.02	0.03	0.42	0.42

It is worth mentioning that the platform on which tests were performed should be upgraded in some places (i.e.

agents should start with an amount of money adequate to the number of stocks which are on the market, genetic algorithm should not create a specified number of new agents as the result of mutation operator after generation, etc.). For the purpose of this test no upgrades were performed (only the mentioned change of the decision algorithm took place). Authors suspect, that even better results could be gained by the use of newer discovering patterns method if some patches to the existing platform were provided. Performed experiment was a first trial of integration and has shown that there is still some place for improvements.

V. Conclusion and Future Plans

The aim of the research presented in this paper was to design of an effective method which is able to properly recognize a given pattern in the time series data. Based on the results of experiments we can draw the conclusion that the proposed method can properly discover the sequences of data within time series. Moreover, when the network is trained the process of recognition is easy and fast. The network response arrives immediately. The only difficulty can be the network training – the choice of appropriate training patterns and the parameters of training, but after some trials and getting more experience this problem disappears.

However the results are promising, there are still improvements possible, for instance other optimization technique of finding the series of shorter or longer widths than the number of input neurons in the network could be proposed. As it was shown *TC* function limits the number of searched widths but it is not the ideal solution, because some proper patterns can be omitted. Some improvements can be made in the test platform, as well. Some upgrades in this system can have an impact on the trustworthy of the performed tests. All mentioned problems and places where improvements can be made are great opportunity to continue studies on the proposed discovering technical analysis pattern method.

References

[1] Fanzi Z., Zhengding Q., Dongsheng L. and Jianhai Y., *Shape-based time series similarity measure and pattern discovery algorithm*, Journal of Electronics (China) vol. 22, no 2 , Springer, 2005.

[2] Fogel G. B., *Computational intelligence approaches for pattern discovery in biological systems,* Briefings in Bioinformatics 9(4), pp. 307-316, 2008.

[3] Fu T., Chung F., Luk R., Ng Ch., *Stock time series pattern matching: Template-based vs. Rule-based approaches,* Engineering Applications of Artificial Intelligence**,** vol. 20, Issue 3, pp. 347-364, 2007.

[4] Guimarães G., *Temporal Knowledge Discovery with Self-Organizing Neural Networks*. *IJCSS*, 1(1), pp.5-16, 2000.

[5] Kwaśnicka *H.* and Ciosmak M., *Intelligent Techniques in Stock Analysis*, Proceedings of Intelligent Information Systems, pp. 195-208, Springer, 2001.

[6] Lee Ch.-H.; Liu A., Chen W.-S., *Pattern discovery of fuzzy time series for financial prediction***,** IEEE Transactions on Knowledge and Data Engineering, vol. 18, Issue 5, pp.613 – 625, 2006.

[7] S., Lu L., Liao G., and Xuan J.: *Pattern Discovery from Time Series Using Growing Hierarchical Self-Organizing Map*, Neural Information Processing, LNCS, Springer, 2007.

[8] Markowska-Kaczmar U., Kwasnicka H., Szczepkowski M., *Genetic Algorithm as a Tool for Stock Market Modelling ,* ICAISC, Zakopane, 2008.

[9] Suh S. C., Li D. and Gao J., *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.