Time-series piecewise evolutionary segmentation based on wavelet transformation and Support Vector Machines

Thomas J. Glezakos[1], Theodore A. Tsiligiridis[1], Spiridon Kintzios[2], Constantine P. Yialouris[1]

[1]Agricultural University of Athens, Department of Science, Informatics Laboratory, 75 Iera Odos, 11855, Athens, Hellas (phone: +30 210 5294176, 182, e-mail:, t_glezakos@yahoo.com, tsili@aua.gr, yialouris@aua.gr)
[2]Agricultural University of Athens, Agricultural Biotechnology Department, Laboratory of Plant Physiology and Morphology, 75 Iera Odos, 11855, Athens, Hellas (phone: +30 210 5294292 e-mail: skin@aua.gr)

Abstract
The present work contemplates the possibility of utilizing wavelets along with evolutionarily produced meta-data so as to enhance the training of Support Vector Machines (SVMs) confronting time-series problems. Long time-series data sets are invariably characterized by multitudes of cases, nonlinearity, and high percentage of noise or fuzziness. Moreover, the existing classification methodologies applied in order to solve such problems provide only a static training approach. The proposed system is used to design the fittest evolutionary segmentation in time series data sets, an approach which is essentially a search in the input plane aiming to interpret the significance by which the various combinations of the input time stamp elements affect the output vector. This was materialized by a genetic algorithm, which audited the training results of a specially constructed SVM and wavelet time series transformation system, promoting the fitter ones through successive generations. In order to evaluate the contribution of wavelet transformation in the enhancement of the produced meta-data, we compared the results to those produced by non-wavelet transformed time series. The results show that the proposed time series piecewise evolutionary segmentation method, when utilizing a wavelet preprocessing procedure, was able to better control the dimensionality, as well as the noise inherent in the initial raw time series information.

Keywords: Genetic Algorithms; Support Vector Machines; Wavelet Transformation.

1. Introduction

In this work time series analysis is utilised in phytopathology, for the identification of two serious plant viruses by means of artificial intelligence methods and processes. The significance of time series analysis has already been stressed out as crucial in many disciplines and data as diverse as energy, finance, econometrics, biology, clinical medicine, meteorology, hydrology and hydraulics, forestry, plant and animal production, agriculture and bio-informatics. Traditionally, the flagship in time series analysis in order to detect trends and patterns hidden under - and producing - the monitored information, has primarily been armed with statistical methods, such as statistical clustering or regression analysis, while more recently the Autoregressive Conditional Heteroscedasticity (ARCH) models, normal or generalized, have appeared as an alternative [1], [2], [17]. Complementarily, a major boost in artificial intelligence is occurring during the recent years, including various and diverse tools. Artificial Neural Network (ANN) models, Support Vector Machines (SVMs) and Genetic Algorithms (GAs) have once again attracted the attention of analysts, experts and consultants, mainly due to the fact that the hardware and know-how we now have at our disposal is capable of simulating such procedures much easier than a few years ago [4], [5], [9], [10], [13], [14], [15], [16].
Our research is focused on engaging a dynamic approach towards applying a piecewise evolutionary segmentation modelling procedure on time series initial information so as to produce fit meta-data for the training of artificial intelligence tools. This implementation, first proposed in [7] and [8], utilized a vibrant segmental methodology to run through the time series adapting the width, the number and the contents of the segments to the data explored. The developed model uses a specially constructed GA in order to search for a feasible solution, which might be the optimal. Each generation of the algorithm produces a population of trainers, each of which maps part of its genome onto the initial time series information, a depiction achieved according to a mechanism dictated by certain genes inside the genome. Thus, a multitude of possible meta-data sets are produced and subsequently used in turn for the training and testing of an ANN. This procedure ultimately produces a fitness score for each meta-data set derived by the performance of the classifier and assigned to the corresponding trainer. The successive generations continue until the best trainer (i.e. the one arousing the best response on the part of the classifier) is discovered. The results showed that after successive generations, in the end the GA revealed the best possible segmentation for the initial time series information. However, this success did not come without a price. The computational cost paid both in time of training and in resources depleted is below today's standards. In particular, the information used for the viruses'

identification is characterized by high non-linearity and non-stationarity and this may therefore in an extent explain the longevity of the training procedure. The whole procedure may be stressed because the ANN is not a fit method to cope with non-stationary data, if effective preprocessing does not take place.
The present work reveals an innovative research regarding the evolutionary production of non time series meta-data and their use in the performance enhancement of a Support Vector Machine (SVM) classifier, as opposed to the constant re-sampling methods used up to now [6]. It also engages wavelet theory to further enhance its results. In other words, in this study ANNs have been replaced by SVMs and the piecewise evolutionary segmentation method has been coupled with wavelet decomposition, in an attempt to enhance the preprocessing of the non-stationary initial time series information. Wavelet decomposition at various scales produces a few coefficients and provides an effective insight into non-stationary time series data.

2. Problem Statement

The basic incentive for the current research emerged on the grounds of phytopathology, specifically in the attempt to identify two serious plant viruses, the *Cucumber green mottle mosaic virus* (GMMV) and the *Tobacco rattle virus* (TRV). In this work, we engage the recently introduced BERA method in order to acquire the initial data set [11], [12]. The products of the method are time series of electrical potential difference, resulting from the virus interaction with properly structured reagents. The two plant virus waves are measured for 331 seconds, for which, at a sampling rate of 10 Hz, the average value was calculated and recorded for each second. Therefore, 331 average values are obtained for one signal channel corresponding to one virus (signature). While reacting to the biosensors, each of the virus waves exhibit characteristic patterns of responses over specific ranges of concentrations. These responses are thus considered as virus features, a real identification signature, which should be analyzed and classified in order to identify the pathogen in question. The data set acquired for the problem consisted of 1,271 instances of time series signatures for both the TRV and the CGMMV. The initial information was split in the training and testing data set consisting of 1079 and 192 instances respectively. The overall data set was balanced for this two-class classification problem, consisting in particular of 640 and 631 instances for the CGMM and TR viruses respectively.
Time series analysis was essential and crucial for the solution of the aforementioned problem. Under this prism, the control of the dimensionality of the input data vector was twofold: Daubechies and Haar wavelet coefficients were passed on to our specially constructed system which searched for the fittest possible meta-data training record set in an evolutionary fashion

3. Materials and methods

3.1. Support Vector Machines

The SVM has been recognised as a highly regarded state-of-the-art classification method. In [18], Boser et al. expressed their maximum margin classifiers theory and developed related training algorithms. The margin is the distance from a hyper-plane separating the classes to the nearest point in the dataset. The advantage of the maximum margin requirement is two-fold: firstly, it produces a solution unique for linearly separable problems and secondly it offers robustness against noise in data. Based on the structured risk minimization (SRM) principle, SVMs are formulated on the basis of minimizing the upper bound of the generalization error.
For the purposes of our research we have used two kernel functions, the PF and the RBF defined as follows:

a. PF Kernel

$$K(\vec{x}_i, \vec{x}_j) = (a + b\vec{x}_i^{\,T}\vec{x}_j)^d$$

In the above the kernel parameters a and b are constants and d is the degree of the polynomial function. A special case of this kernel $a = 0, b = d = 1$ forms a linear kernel, i.e. LF: $K_L(\vec{x}_i, \vec{x}_j) = \vec{x}_i^{\,T}\vec{x}_j$.

b. RBF Kernel

$$K(x_i, x_j) = \exp(-\gamma \left\| x_i - x_j \right\|^2), \gamma > 0$$

The kernel width common to all the kernels is specified a priori by the user. For the purposes of our research we used the C-SVM and SVR types of support vector machines provided by the libSVM tool [3] and utilizing two kernels, the PF and the RBF.

3.2. Genetic Algorithms

GAs on the other hand, are inspired by evolutionary biology, and especially driven by the Darwinian axiom of the "survival of the fittest", incorporating numerous biological procedures such as inheritance, selection, crossover

(recombination) and mutation. They are mostly implemented as computer simulations which search the input plane for 'better' solutions to a given optimization problem. The GA starts out with an, often random, initial population of encoded representations of candidate solutions. These representations are referred to as chromosomes, genotypes or genome, while the candidate solutions are referred to as phenotypes. The algorithm proceeds by generations each of which is comprised by genotypes of the previous generation, which are elected basically by their fitness and modified on the grounds of a possible recombination or mutation to form a new population of higher overall expected fitness.

3.3. Wavelet analysis

The concept of wavelets is a relatively new approach to signal processing, allowing for time series signals transformation in the time and frequency domain simultaneously. By decomposing the signal to its wavelets, this transformation often over-powers Fourier transformation in many aspects, including the analysis of even non-stationary signals. Wavelet processing decomposes the signal producing much smaller derivative coefficients / approximations, which are easier to manipulate and also caters for the final reconstruction of the original signal. To achieve this task, wavelet analysis proceeds as follows:
Let *x(t)* a time series signal, then the Continuous Wavelet Transformation (CWT) is defined as

$$CWT_x^y(\tau,s) = \Psi_x^{\psi}(\tau,s) = \frac{1}{\sqrt{s}}\int_{-\infty}^{+\infty} x(t)\Psi^*(\frac{t-\tau}{s})dt$$

The above transformation produces a signal which is dependent on two main factors, s and τ, the translation and scale parameters respectively. The concept of scale in wavelet analysis replaces the concept of frequency used in Fourier transformation and is essentially a 'zooming' factor to the signal, i.e. taking a higher resolution snapshot of the oscillation. Higher scales (lower frequencies) correspond to a wider view of the signal, whereas lower scales to a more detailed view. *Ψ(t)* is the transforming function, often referred to as the "mother wavelet", due to the fact that it is an oscillatory (wave) window function of finite length (small) running through the signal (i.e. wavelet). The translation term (*τ*) denotes the position of the transformation window, as it runs through the signal and it obviously relates to the time information of the transformation process. The CWT requires large amounts of computation time and resources, while the Discrete Wavelet Transformation (DWT) is considerably simpler to implement while drastically reducing computational needs. DWT uses a fixed scale throughout as a dilation step, making the mother wavelet a discrete function in the sense that the time-space scale is furthermore sampled at discrete intervals. In this work DWT scale and translation were chosen such that the sampling of the scale (frequency) and the time axis was dyadic (i.e. s=2, t=1).

4. Developing the system

4.1. Layout of the Algorithm

The developed system bears simple physiognomy but complex mechanics, offering the possibility to load a vast number of training data, practically covering every width and length of time series information. Upon initiating the procedure, the system will generate the first generation of trainers and map their genes to the time series producing the first generation of meta-data. These will be thrown into the artificial intelligence tool, defined by the SVM in this case [3], while the most efficient record sets will emerge in the next generation. The meta–data production module is derived through the implementation of a GA, which proceeds as prescribed in the following pseudocode.

GA pseudo code for the production of fit meta-data

```
Set the ideal accuracy, trainer population, recombination and mutation probabilities, maximum training epochs
Select the time series train and test data
Construct basic SVM module
WHILE SVMAccuracy < idealAccuracy:
    BEGIN
        Assembly the generation's trainer population
        For each individual trainer:
        BEGIN
            map trainer chromosome to ts data
            Train and test SVM
            Derive SVMAccuracy
            Calculate trainer fitness score
            Assign fitness to chromosome
```

END
Select trainers on fitness score
Store best trainers
Formulate next generation of trainers
Apply recombination and mutation
END
Save best derived SVM

The algorithm initiates by producing a random population of trainers in its initial generation. These are binary vectors, which are considered as chromosomes behaving in certain predefined ways and able to appropriately manipulate the initial raw time series. Each successive trainer is set to map its genes to the initial raw time series data, thus producing a meta-data time series recordset, which is used so as to train and test the SVM classifier. After the testing phase, each trainer is assigned a fitness value, i.e. the distance of the accuracy of the corresponding SVM from the ideal accuracy set in the initialization phase. Thereinafter, the next and subsequent generations of the algorithm are formulated according to a selection policy which elects the fittest members of the previous generation of trainers. The most potent and powerful trainers, which are more likely to be selected for the next generation, are the ones producing the highest accuracy. Thus, after a number of generations, the algorithm is expected to create a powerful SVM object, of an almost ideal accuracy percentage.

4.2. The Structure of the Trainer Population

The first and subsequent generations of the GA start out by assembling a population of a user defined number of chromosomes, the genome of which consists of randomly chosen binary genes (0 or 1). Each chromosome is used so as to manipulate the input raw time series data according to their structure, in order to produce fitter educational meta-data sets used in the supervised learning of the neural network. The genome of the chromosome is divided in two parts: the first part consists of a number of randomly chosen binary bits, equal to the time fragments of the initial raw time series data and is called the activation part for it bears as its duty to carry out the instructions given by the behavioral mechanism. The second part of the chromosome, located at the beginning of the genome, is the behavioral core mechanism of the chromosome. It consists of two supplementary random binary bits, the structure of which defines a set of rules deciding the actions taken by the chromosome towards the raw data that it manipulates.

4.3. Data Manipulation and Chromosome Mapping

Each training chromosome exists in order to map its genes to the raw data set, according to the structure of its core mechanism. Thus, a number of meta-data sets equal to the user defined number of chromosomes for each generation of the algorithm are produced and used for the training of the neural network. The mapping of the chromosome onto the raw data set is essentially one of a number of descriptive statistics functions, dictated by the behavioral core mechanism of each chromosome, which is responsible to manipulate the initial raw time series and produce meta-data in a rational way.
Thus, essentially four behaviors have been chosen, including either resampling procedures or grouping ones. One method resamples data, while the rest design groups of data in the initial time series and produce a unitary number for each group, which also stands out as a memory for the segmentation of the time series. The only parameter which varies in these three behaviors is the essence of this memory. In the first case, we represent each designed segment by the average of its data, while in the second, we divide it in two equal parts, taking the exact middle value. Finally, with the last mechanism we estimate the essential width of each designed segment, embedding this aspect also into our system. Specifically,

- If the core mechanism genes are “00”, then this stands out as a “Discard-All-Zeros” resampling function. In this case, the time series will be stripped off of its values for which the corresponding genes of the trainer is 0.
- If the core mechanism genes are “11”, then this stands out as a “First-One-Last-Zero Average” piecewise segmentation mechanism for the initial data. In this case, the trainer extracts the average of the time series elements for every group of its own genes which start with the first 1 and end with the last 0.
- In the case in which the core mechanism is “01”, the chromosome behaves as a “First-One-Last-Zero median” piecewise segmentation mechanism, which returns the median of the initial data series elements for each segment which corresponds to the first 1 and the last 0 of its own genes.
- Finally, if the core mechanism is “10”, then the chromosome will return the distance of the maximum to the minimum value of every group of the initial time series which is defined in the same aforementioned manner.

During the mapping procedure the algorithm examines the core mechanism genes of each trainer in the row and acts accordingly. Table 1 provides an example of meta-data produced from time series data, according to action taken by the mechanism genes. For example, if the core mechanism genes are “00”, then we have a straight forward re-sampling behavior. The meta-data produced (44, 8, 8, 18, 48) are the only time fragments of the initial time series for

which the genes of the trainer chromosome are 1. In the second case, where the core mechanism genes are "11", then piecewise segmentation and averaging in each segment occurs. In this case, the chromosome defines five segments in the time series, as follows:

- Segment 1: elements 1,2,3. Average(44,32,17)=31
- Segment 2: element 4. Average(8)=8
- Segment 3: elements 5, 6, 7. Average(8,12,1)=7
- Segment 4: elements 8, 9. Average(18,30)=24
- Segment 5: element 10. Average(48)=48

Table 1. Example of the mapping of chromosomes onto raw initial time series data, according to their core behavioral genes

Initial raw time series										
	44	32	17	8	8	12	1	18	30	48
Chromosome genome										
	1	0	0	1	1	0	0	1	0	1
Core Genes	**Meta data production**									
00	44			8	8			18		48
11	31			8	7			24		48
01	32			8	8			24		48
10	27			8	11			12		48

4.5. The survival of the fittest

For each generation of the algorithm, each trainer is assigned a fitness score which is derived by the trainer's performance assessed by the accuracy of the SVM object. This is essentially the proximity of the potential of the trainer to an optimal solution set in the initialization of the process. Let r_{ij} be the accuracy value of the SVM trained and tested with the meta-data produced by the *i-th* trainer of the *j-th* population of the algorithm. Then, the fitness score f_{ij} assigned to the *i-th* trainer of the *j-th* population, should be $f_{ij} = 1/|r_{ij} - h|$, where h is the accuracy threshold, which maximizes the f_{ij}. As f_{ij} rises in value, r_{ij} moves closer to h and our system crawls nearer to the ideal solution h arbitrarily set in the beginning of the algorithm. Maximizing f_{ij} produces stronger and more potent populations of trainers. Note that the accuracy of the SVM object is measured during the testing phase, where it is applied on totally unseen cases, i.e. the evaluation meta-data set.

The algorithm stores the meta-data produced, and then updates each trainer with the corresponding fitness score. The policy of selecting the best offspring incorporated the stochastic procedure known as roulette wheel selection. Thus the fitness score f_{ij} of the *i-th* trainer (i = 1,2,…,V) of the *j-th* population (j=1,2,…,M) has probability $p_{ij} = f_{ij} \Big/ \sum_{i=1}^{V} f_{ij}, \forall j = 1,2,...,M$ of being selected.

The roulette wheel incorporates a fitness proportionate selection operator, which elects to perpetuate the fittest chromosomes, i.e. the ones with the higher fitness score. In this context, chromosomes with relatively high fitness scores are less likely to be eliminated. On the other hand, less fit chromosomes may not be extinguished from the genetic pool of the next generations. This results in the fact that some weaker solutions to the problem at hand may survive the algorithm sweep for the forming of the next generations, conveying their potentially useful genes to their offspring.

The algorithm then turns into a decision branch: either the trainers pool is empty, so it is assumed that this is the first generation and so produces the initial population, or there are trainers present with fitness scores assigned to them, so it proceeds to mating, crossover and mutation in order to formulate the next generation of trainer chromosomes. Recombination and mutation is then performed under user defined probabilities on the next generation of trainer chromosomes, thus allowing the algorithm to start processing a new generation of meta-data.

5. Results

As a case study, we seek to identify two plant viruses, namely the *Tobacco rattle virus (TRV)* and the *Cucumber green mottle mosaic virus (CGMMV)*, by examining their responses towards certain prescribed reagents. By reacting

to these biosensor reagents, each virus produces unique patterns of biosensor response in relation to the reagent concentration. This set of responses in the time unit forms a special signature for each virus, a time series data set, the examination of which becomes fruitful for its identification. The developed system, equipped with the SVM identification machine as provided by the libSVM library, was trained using the dataset described in the section 2. Using the parameterization referred to in Table 2, we were able to perform the evolutionary production of meta-data from the initial time series information and to train and test the corresponding SVM.

Table 2. System parameters selected for the virus identification case

Parameter	**Value**	
SVM Type	C-SVC	
SVM kernel	RBF	Γ=adaptive
	Polynomial	D=3
C parameter	150	
Trainer population	25	
Generations	100 max	
Selection method	Roulette wheel	
Mutation rate	5‰	
Crossover rate	40%	

The first time series subset underwent a 10-fold cross validation procedure by the system in order to decide on the best C parameter. The validation set was used in order to derive the system prediction accuracy, which was elected to be the ratio of the correctly classified patterns to the overall number of patterns in the validation data set. After grid-search cross validation, we chose $C = 150$ for all our tests. The degree d of the PF, as well as the width γ of the RBF kernel, decide the flexibility of the resulting classifier. The lowest degree polynomial ($d = 1$) forms the linear kernel. Since there is a sheer increase in complexity with small increases of d we intuitively chose $d = 3$. As for γ, this affects the smoothness of the curvature of the decision boundary, producing smooth surfaces at lower values, increasing their complexity as the values rise. An interesting remark is to look more closely on the relationship between the values of γ and the number of groups designed by the trainer chromosome of the GA. Thus, given that a certain genetic trainer dictates *k* groups in the time series data set, then the decision boundary of the corresponding SVM in the current generation will have a curvature decided by $\gamma = 10^{-3} k$.

5.1. Meta-data production without wavelet transformation

Table 3 lists the γ parameters for the five best performing trainer chromosomes as resulted from the application of our system on the virus identification problem. Each constructed trainer was monitored by the SVM as regards to its accuracy towards the evaluating data set. In this context, we utilized and tested the RBF and the PF kernel functions in the classification mechanism of the proposed system, in order to assess their potential. In the course of our testing experiments, the RBF kernel machine proved to be more powerful than the polynomial one, achieving a classification percentage of 96.88% in the 92nd generation and thus outperforming the PF kernel, which was confined to a mere 85.42% in the fifth generation. Table 3, apart from the *k* and *γ* parameters of the RBF kernel, also depicts the five best accuracy percentages achieved by the two kernel machines, along with the generation where this was achieved.

Table 3. Five best accuracy percentages for the RBF and the Polynomial kernel

SVM with RBF kernel				**SVM with Polynomial kernel**	
Gen.	**Accuracy (%)**	**k**	**γ**	**Gen.**	**Accuracy (%)**
92	96.88	172	0.172	5	85.42
56	96.35	167	0.167	4	84.90
17	95.83	165	0.165	7	84.38
43	95.31	173	0.173	8	83.85
9	94.79	159	0.159	14	83.33

As shown on Table 3, as well as on the following Figure 1, the evolutionary process achieves its best results faster with the PF kernel rather than with the RBF one, but the polynomial SVM is not further optimized. On the contrary, the RBF kernel reaches its best performances later on, but these are much more potent, achieving a score of only 3% false classification score in the 92nd generation. It is also clear that the system starts out low for both kernels. The average accuracy of the first six generations does not supersede the score of 60%, but from the ninth generation on, the accuracy of the system stabilizes at around 78% for the PF and 88% for the RBF kernel.

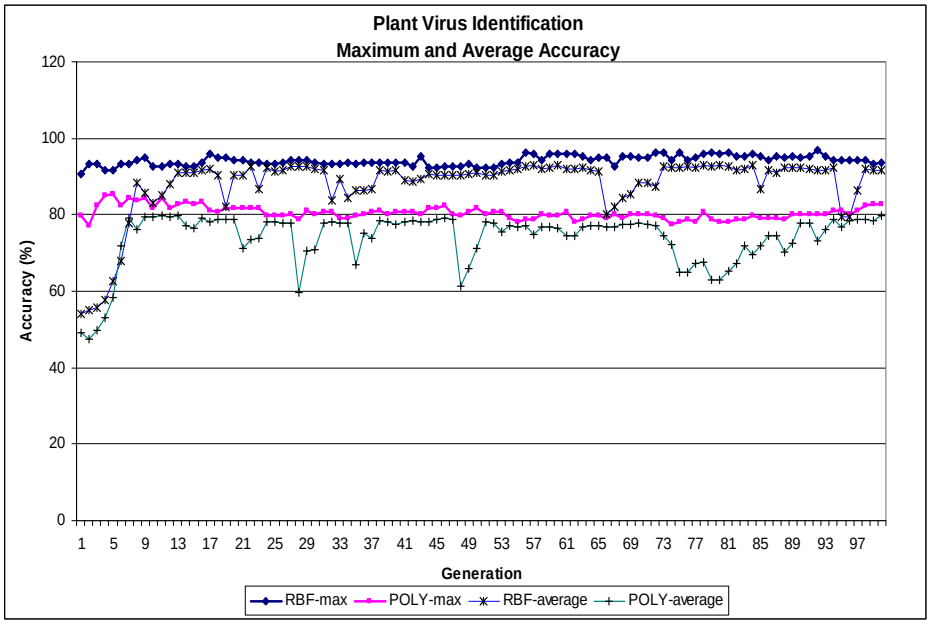

Figure 1. RBF and PF kernel performance for the plant virus identification problem

By applying an evolutionary piecewise segmentation scheme on the initial raw time series information, the system managed to produce a series of meta-data which were getting fitter after successive generations, before reaching a final accuracy plateau. It was made clear by this enhancement that the initial time series information was simplified to its essential parts, providing more robust piece of training material for the classifier. What was more interesting at this moment was the effect that wavelet transformation might impose on the time series, before the latter passed through the evolutionary procedure.

5.2 Meta-data production with DWT preprocessing

In this scenario DWT was used as a preprocessing technique, before the time series was passed to the piecewise evolutionary segmentation method. Among the multitudes of wavelet families in existence we elected Daubechies and Haar first level wavelet coefficients.

Table 4. Five best accuracy percentages for the SVM and the polynomial kernel when DWT was performed before the piecewise evolutionary segmentation.

Db-10 Polynomial		Db-10 RBF		Haar Polynomial		Haar RBF	
Generation	Accuracy	Generation	Accuracy	Generation	Accuracy	Generation	Accuracy
1	97.39	94	95.31	14	97.39	140	91.15
17	96.87	53	94.79	1	96.87	4	90.62
81	96.35	90	94.27	5	96.35	2	89.58
29	95.83	28	93.75	4	95.83	46	89.06
69	95.31	15	93.23	-	-	1	88.54

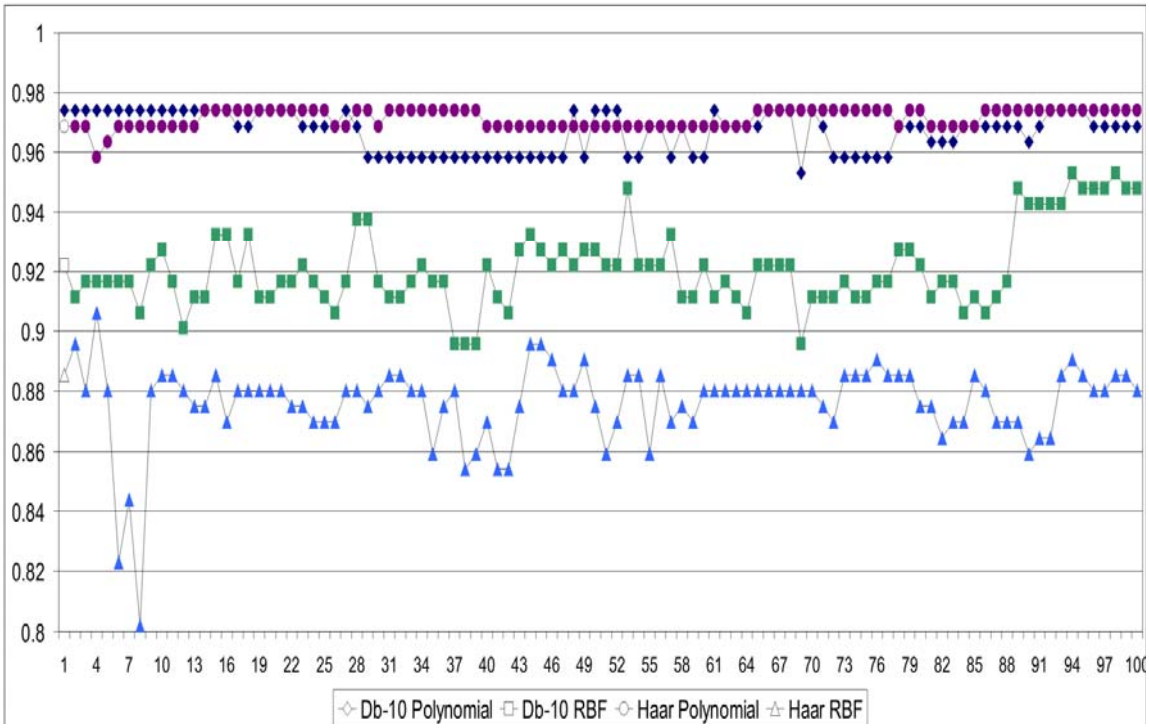

Figure 2 RBF and PF kernel performance when DWT was performed before the evolutionary piecewise segmentation

The results depicted in Table 4 and Figure 2 reveal that in this case the PF kernel is much more capable than the RBF one. It not only responds better to all the prism of the evolutionary segmentation method, but it also cuts out the time cost of the procedure arriving at its best accuracy early on for the Db-10 and Haar coefficients – i.e. 1st and 14th generation respectively with an accuracy of 97.39% for both.

6. Conclusions

Time series information often arise while monitoring various phenomena, including environmental and agricultural processes, the management of which arises strict demands for credible and accurate forecasting and classification potential. Time series analysis functions formulate a sequence which bears the distinctive marks of a certain internal structure. This is generated by factors that influence the generation of the time sequence values over time and could be analyzed in two main components: trend and seasonality. Thus, time series analysis conforms so as to obtain an understanding of the underlying forces which produced the observed information and ultimately develop a model by which to classify and forecast a certain procedure. Up till now, time series analysis mainly employs statistical re-sampling or moving average methods in an attempt to smooth out the negative effects of random variation. The estimation of the optimal width of the smoothing technique is of crucial importance as it defines the amount of information manipulated. Being a factor heavily dependent on the nature of the problem, the moving width is traditionally defined after trial and error in a constant fashion, i.e. it runs through the time series without altering its value.

Through the current research we show that it is possible to use a dynamic smoothing factor for the definition of an optimal piecewise segmentation procedure. The algorithm developed is able to produce non time series meta-data in order to be incorporated in an evolutionary process along with the input of time series models. The results strength and facilitate the learning of predictive artificial intelligence tools. Using the combined power of GAs and SVMs, the proposed system utilizes an interchangeable variation of a vibrant piecewise segmentation method to smooth out possible inhibitory aspects of time series information, adjusting its behavior according to the data explored. The developed tool has so far been applied on two cases, covering the categories of classification and regression problems. From the displayed performance and taking into account that the results were recorded during the testing phase of the classifier, we can deduce that the proposed technique yields better performance as regards to classification than to regression problems. However, and taking into account that the RBF kernel has been proven generally more powerful for our data than the PF kernel, in both cases we were able to apply an evolving genetic

data segmentation procedure on the initial data set and note on the enhanced behavior of the system, which was able to control the dimensionality and the noise of the input vector, better in the former case and acceptably in the later. The plans of the research team for the near future include the enhancement of the tool with modules to genetically adapt its structure – i.e. kernel function, hyper-parameters - to the data given. Also, we seek to conduct a thorough comparison of this tool with neural and RBF-networks, applying them to problems originating from the agricultural and the environmental sector.

References

1. Bali, T.G., 2008. The intertemporal relation between expected returns and risk. Journal of Financial Economics 87 101–131.
2. Bollerslev, T., 1986. Generalized Autoregressive Conditional Heteroscedasticity. Journal of Econometrics 31 307–327
3. Chang, C.-C., Lin, C.-J., 2001. LIBSVM : a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.
4. Chang, C.L., Lo, S.L., Yu, S.L., 2005. Applying fuzzy theory and genetic algorithm to interpolate precipitation. Journal of Hydrology 314 92–104.
5. Chau, K.W., 2007. A split-step particle swarm optimization algorithm in river stage forecasting. Journal of Hydrology 346 131–135.
6. Frossyniotis, D., Anthopoulos, Y., Kintzios, S., Perdikaris, A., Yialouris, C.P., 2006. A multisensor fusion system for the detection of plant viruses by combining Artificial Neural Networks. In: Kollias, S.; Stafylopatis, A.; Duch, W.; Oja, E. (Eds.), 16th International Conference on Artificial Neural Networks. ISBN: 978-3-540-38625-4, Springer: Berlin
7. Glezakos, T.J., Moschopoulou, G., Tsiligiridis, T.A., Kintzios, S., Yialouris, C.P.: Plant virus identification based on neural networks with evolutionary preprocessing. Computers and Electronics in Agriculture, Volume 70, Issue 2, March 2010, pp. 263-275 (2010)
8. Glezakos, T.J., Tsiligiridis, T.A., Yialouris, C.P., Maris, F., Ferentinos, K.P.: Feature Extraction for Time Series Data: an Artificial Neural Network Evolutionary Training Model for the Management of Mountainous Watersheds. Neurocomputing, Volume 73, Issues 1-3, December 2009, pp. 49-59 (2009).
9. Jain, A., Kumar, A.M., 2007. Hybrid neural network models for hydrologic time series forecasting. Applied Soft Computing 7 585–592.
10. Kerh, T., Lee, C.S., 2006. Neural networks forecasting of flood discharge at an unmeasured station using river upstream information. Advances in Engineering Software 37 533–543.
11. Kintzios, S., Bem, F., Mangana, O., Nomikou, K., Markoulatos, P., Alexandropoulos, N., Fasseas, C., Arakelyan, V., Petrou, A-L., Soukouli, K., Moschopoulou, G., Yialouris, C., Simonian, A., 2004. Study on the mechanism of Bioelectric Recognition Assay: evidence for immobilized cell membrane interactions with viral fragments. Biosensors & Bioelectronics 20 907-916.
12. Kintzios, S., Goldstein, J., Perdikaris, A., Moschopoulou, G., Marinopoulou, I., Mangana, O., Nomikou, K., Papanastasiou, I., Petrou, A-L., Arakelyan, V., Economou, G., Simonian, A., 2005. The BERA Diagnostic System: An all-purpose cell biosensor for the 21th Century. 5th Biodetection Conference, Baltimore, MD, USA.
13. Kuncheva, L., 2000. Combining Classifiers by Clustering, Selection and Decision Templates. Technical report. University of Wales, UK.
14. Liu, F., Ng G.S., Quek, C., 2007. RLDDE: A novel reinforcement learning-based dimension and delay estimator for neural networks in time series prediction. Neurocomputing 70 1331-1341.
15. Ni, J.R., Xue, A., 2003. Application of artificial neural network to the rapid feed back of potential ecological risk in flood diversion zone. Engineering Applications of Artificial Intelligence 16 105–119.
16. Recknagel, F., 2001. Applications of machine learning to ecological modelling. Ecological Modelling 146 303–310.

17. Yamaguchi, K., 2008. Reexamination of stock price reaction to environmental performance: A GARCH application. Ecological Economics 68 345–352.

18. Boser, B.E., Guyon, I.M., Vapnik, V.N., 1992. A training algorithm for optimal margin classifiers, In Hausler, D. (Ed.) Proceedings of the Fifth Annual Workshop on Computational Learning Theory. 144-152, Pittsburgh, PA, ACM Press.