Improving Time Series Classification Using Hidden Markov Models

Bilal Esmael
University of Leoben
Leoben, Austria
Bilal@stud.unileoben.ac.at

Arghad Arnaout
TDE GmbH
Leoben, Austria
Arghad.Arnaout@tde.at

Rudolf K. Fruhwirth
TDE GmbH
Leoben, Austria
Rudolf.Fruhwirth@tde.at

Gerhard Thonhauser
University of Leoben
Leoben, Austria
Gerhard.Thonhauser@unileoben.ac.at

***Abstract*- Time series data are ubiquitous and being generated at an unprecedented speed and volume in many fields including finance, medicine, oil and gas industry and other business domains. Many techniques have been developed to analyze time series and understand the system that produces them. In this paper we propose a hybrid approach to improve the accuracy of time series classifiers by using Hidden Markov Models (HMM). The proposed approach is based on the principle of learning by mistakes. A HMM model is trained using the confusion matrices which are normally used to measure the classification accuracy. Misclassified samples are the basis of learning process. Our approach improves the classification accuracy by executing a second cycle of classification taking into account the temporal relations in the data. The objective of the proposed approach is to utilize the strengths of Hidden Markov Models (dealing with temporal data) to complement the weaknesses of other classification techniques. Consequently, instead of finding single isolated patterns, we focus on understanding the relationships between these patterns. The proposed approach was evaluated with a case study. The target of the case study was to classify real drilling data generated by rig sensors. Experimental evaluation proves the feasibility and effectiveness of the approach.**

Index Terms – Hidden Markov Model, Machine learning, Time series Classification.

I. Introduction

Time series data are ubiquitous and broadly available in a broad range of applications in almost every domain. To gain knowledge from these data, numerous clustering and classification methods were developed [1]. Time series classification is a supervised machine learning problem aimed for labeling multivariate series of variable length [2]. It is an elementary procedure enables us to easily monitor the systems and detect the events (activities) that have been taken place during the whole process.

Time series data often have a very high dimensionality. Thus, classifying such data poses a challenge because a vast number of features can be extracted [4][9]. That means, applying classification techniques directly on raw time series data is often not practical. To overcome this problem, the original raw data are usually replaced by a higher-level representation. This representation allows extracting higher order features [1][3].

Window-based representation is a common representation in machine learning. Some examples of such techniques are Piecewise Aggregate Approximation (PLA) [10], Trend-based and Value-based Approximation (TVA) [9], and Symbolic Aggregate Approximation (SAX) [4]. In window-based representation, the whole time series data are divided into a sequence of equal sized windows (segments). One feature or more are extracted from each frame, and a vector of these features becomes the data-reduced representation.
For time series classification, the created vectors are used to train a classifier. This classifier could be Support Vector Machine (SVM), Neural Network (NN), Rule Induction or any other classifier. In above mentioned classification approach, the temporal relations existing in time series data are ignored, which often leads to inaccurate classification results.

In this work we propose a multiple classifier approach to improve the accuracy of time series classifiers. The improvement is performed by adding a HMM model to the classification system. The added model will regard the temporality of the data, and execute a second stage classification. Multiple classifier systems are more promising than those of single classifier systems for a wide range of applications [3][6][7]. The "No Free Lunch" Theorem states that there is no single learning algorithm that in any domain always induces the most accurate learner [3]. Therefore, the objective of this approach is to utilize the strengths of one technique to complement the weaknesses of another [6].

In this work, we suppose that the system which generates the data works under a number of latent states as described in Fig. 1. Many time series datasets satisfy the mentioned assumption. For example, in oil and gas drilling systems all mechanical parameters e.g., rotary speed and hook load, are continuously measured by a group of rig sensors. In addition, these systems move dynamically from a state to another.

The remainder of the paper is organized as follows: Section II explains the basics and advantages of Hidden Markov Models. Section III presents the general framework of our approach. Sections IV and V show the details of the training and classification algorithms. Section VI presents the evaluation procedure and the experimental results.

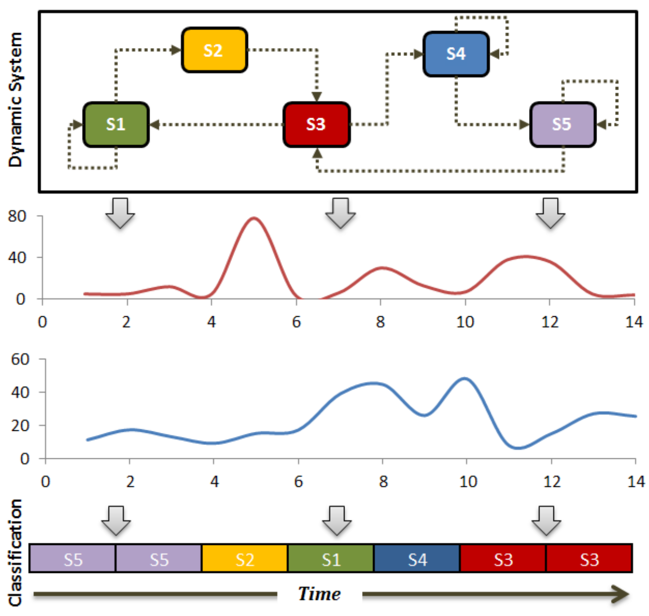


Figure 1. Dynamic Systems: The objective of this approach is to classify the time series data generated by state-machine based systems.

II. Hidden Markov Model

A Markov chain is a mathematical model that consists of a finite number of states $\{S_1, S_2, \dots, S_n\}$ and some known probabilities $P = \{p_{ij}\}$ where p_{ij} is the probability of moving from state S_i to the state S_j. Markov chain has the Markov property (memoryless) where the next state depends only on the current state and not on the sequence of states that preceded it [5]. In other words, the Markov chain should satisfy equation 1.

$$P(S_{t+1}|S_0, S_1, \dots, S_t) = P(S_{t+1}|S_t) \qquad (1)$$

That means, the state at any given time $t + 1$ depends only on the previous states at time t.

Hidden Markov model is a Markov chain in which the states are not directly visible, but observations, dependent on the state, are visible. Each state has a probability distribution over the possible output observations. Therefore the sequence of observations generated by HMM model gives information about the corresponding sequence of states. The “hidden” word in Hidden Markov Models means that the observer does not know in which state the system may be in, but has only a probabilistic insight on where it should be. Hidden markov model is usually defined by the following equation 2. The structer of HMM is described in Figure 2.

$$\mu = (S, O, A, B, \pi) \qquad (2)$$

Where

- The variable $S = \{s_1, s_2, \dots, s_n\}$ is the hidden states set of the model.
- The variable $O = \{o_1, o_2, \dots, o_m\}$ is the observations set of the model.
- The variable A is the transition matrix. It is given by equation 3.

$$A = \{a_{ij}\}, \qquad a_{ij} = P(s_j|s_i) \qquad i, j = 1,2, \dots, n \qquad (3)$$

- The variable B is the emission matrix. It is given by equation 4.

$$B = \{b_{ij}\}, \qquad b_{ij} = P(o_j|s_i) \qquad i = 1,2, \dots, n \qquad j = 1,2, \dots, m \qquad (4)$$

- The variable π is the initial state matrix where π_i is the probability that state s_i is a start state.

Hidden Markov Models are used to model temporal and sequence data. They are commonly used in temporal pattern recognition such as time series classification [8], speech recognition [5], part-of-speech tagging, and bioinformatics.

There are three basic problems associated with hidden Markov models. These problems are the following:

A. *The Learning Problem*

Basically, the learning problem is how to find the parameters of a HMM model $\mu = (A, B, \pi)$ given a sequence of observations (training data) $O = \{O_1, \dots, O_T\}$. The target of the learning problem is to maximize $P(O|\mu)$. This problem can be solved using Baum–Welch algorithm

B. *The Evaluation Problem*

Given the parameters of the model $\mu = (A, B, \pi)$, and a sequence of observation $O = \{O_1, \dots, O_T\}$. How to compute the probability of the observation sequence $P(O|\mu)$.

C. *The Decoding Problem*

Given the parameters of the model $\mu = (A, B, \pi)$ and a particular output sequence $O = \{O_1, \dots, O_T\}$. How to choose a corresponding state sequence $S = \{S_1, \dots, S_T\}$ that is most likely to have generated that output sequence. This requires finding a maximum over all possible state sequences. This problem can be solved efficiently using Viterbi algorithm.

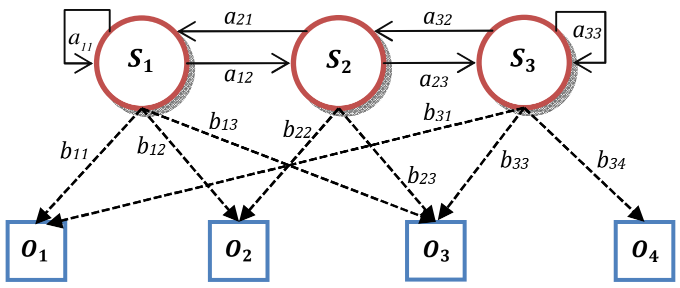


Figure 2. Hidden Markov Model: States =$\{S_1, S_2, S_3\}$, A is the transition matrix, Observations = $\{O_1, O_2, O_3, O_4\}$, B is the emission matrix

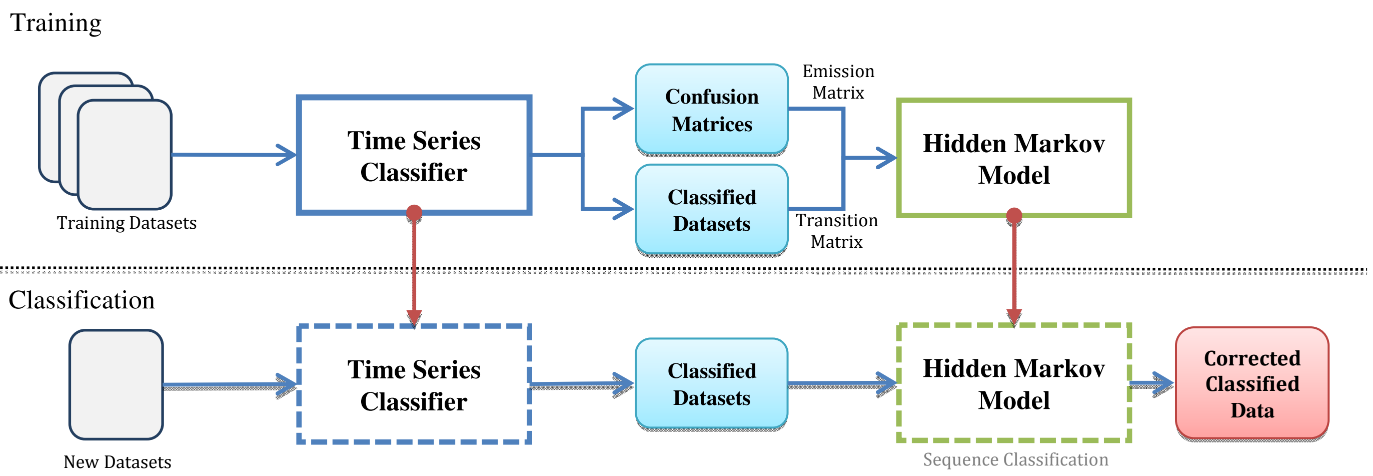
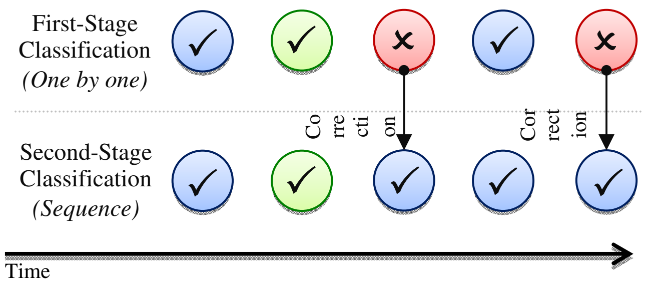


Figure 3. The general framework. There are two phases: Training and Classification

III. The General Framework

The main objective of the proposed approach is to improve the accuracy of time series classifier. To this end a HMM model was trained to evaluate, confirm and correct the classification results performed by the initial classifier. The general framework of the approach is sketched in Fig. 3 which shows the two phases of this approach.

In training phase, the confusion matrices and the classified data outputted by the classifier will be used to train the model. In classification phase, the trained model will be used to reclassify the sequence of classified samples. The correct samples will be confirmed and the misclassified samples will be corrected. In other words, classifying any new dataset will be done using two classification stages as described in Fig. 4. In the first stage, the input data will be classified by the first classifier (sample by sample). In the second classification stage, the classified samples will be grouped as a sequence and reclassified using HMM model.
The HMM model can correct the misclassified samples because it being learned from the mistakes produced by the initial classifier. In addition, it models the dynamic behaviour of the underlying system.



First-Stage Classification (One by one)
Second-Stage Classification (Sequence)
Co rre cti on
Cor rect ion
Time

Figure 4. Sequence Classification

The idea behind our approach is to look at the classification process itself as a dynamic system with hidden states and observations. HMM technique can be used to model such classification system because the output of this system is a sequence of observations, and these observations depend probabilistically on the internal states. In our case the states and the observations are the same.

IV. Training Algorithm

Training a HMM with the original and the classification systems requires calculating the model parameters. Fig. 5 shows the main steps of the training algorithm and further described below.

```
Training Algorithm:
Input
  • A group of classified datasets with their confusion Matrices
Output
  • A trained HMM $\mu = (A, B, \pi)$ where:
      • A is the Transition Matrix,
      • $B$ is the Emission Matrix and
      • $\pi$ is Initial Matrix
Do
  • Create a HMM $\mu$ with $N$ states.
  • Calculate the transition matrix $A = \{a_{ij}\}$ where $a_{ij} = P(S_j|S_i)$ the propability of moving from state $S_i$ to state $S_j$
  • Calculate the emission matrix $B = \{b_i(k)\}$ where $b_i(k) = P(O_k|S_i)$ which is the propability that state $S_i$ emits the observation $O_k$
  • Calculate the initial matrix $\pi = \{p_i\}$ where $p_i = P(S_i)$
  • Add $A$, $B$ and $\pi$ to the model $\mu$
  • Return $\mu$
End
```

Figure 5. The Training algorithm: Training the HMM requires calculating the transition matrix, emission matrix and initial matrix

A. *Transition Matrix A*

The transition matrix A is calculated by the following algorithm (Fig. 6).

```
Transition Matrix
Input
  • A training dataset $D$
Output
  • Transition Matrix $A$
Do
  • Extract a unique list of predefined classes $\mathrm{S} \in \{\mathrm{S}_1, \dots, \mathrm{S}_\mathrm{n}\}$.
    S will be considered as the states set
  • Extract the list of the labels LS from the datasets:
        $LS = \{s^1, s^2, \dots, s^T\}$,    $T$ is length of $D$
                                          $s^i \in S \ \forall \ i = 1, \dots, T$
  • Calculate the transition matrix $A = \{a_{ij}\}$
        $a_{ij} = \frac{(\sum S_i \rightarrow S_j) \times 100}{\sum_{t=1}^{T} s^t = S_i}$    $s^t = S_i, s^{t+1} = S_j$    (5)
  • Return $A$
End
```

Figure 6. Calculating Transition Matrix A

B. *Emission Matrix B*

The emission matrix B is calculated from the confusion matrix C (Table 1).

$$C = \{c_{ij}\}, \qquad i,j = 1,2,..,n$$

The item c_{ij} represents the number of samples that belong to state S_i but they have been classified as S_j. These samples called misclassified samples. Fig. 7 shows the algorithm.

```
Emission Matrix
Input
  • A confusion matrix $C$
Output
  • Emission Matrix $B$
Do
  • Create a matrix B where,
        $b_{ij} = \frac{c_{ji} \times 100}{\sum_{n}^{i=1} c_{ji}}$    $i,j = 1,2, \dots, n$    (6)
  • Return $B$
End
```

Figure 7. Calculating Emission Matrix B

C. *Initial Matrix* $\boldsymbol{\pi}$

The initial matrix is calculated by equation 7. All states have the equal probabilities to be the initial state

$$p_i = \frac{1}{\sum S_i} \qquad i = 1,2,..,n \qquad (7)$$

TABLE. 1 CONFUSION MATRIX

		Actual States					
		S_1	S_2	…	S_j	…	S_n
Predicted States	S_1	c_{11}	c_{12}	…	c_{1j}	…	c_{in}
	S_2	c_{21}	c_{22}	…	c_{2j}	…	c_{2n}
	…	…	…	…	…	…	…
	S_i	c_{i1}	c_{i2}	…	c_{ij}	…	c_{in}
	…	…	…	…	…	…	…
	S_n	c_{n1}	c_{n2}	…	c_{nj}	…	c_{nn}

V. Classification Algorithm

The main difference between traditional machine learning classifier (i.e. SVM, NN, etc.) and Hidden Markov Model is that HMM classifies the data on the basis of their temporal relations. The classification algorithm takes a sequence of observations (labels) as input, and returns a sequence of states as output. In other words, to classify a given sequence of observations, we need to find the most likely sequence of states (path) that produces these observations. Viterbi algorithm [11] was used to find the most likely path through a trellis. The trellis in our case is a graph of a states set and observations. Each node in trellis represents a state and each edge represents a transition between two states. An example of a trellis is shown in Fig. 9. The pseudocode of the classification algorithm is shown in Fig. 8.

```
Classification Algorithm
Input
  • The trained HMM $\mu = (A, B, \pi)$
  • A sequence of observations $O = \{o^1, \dots, o^T\}$
Output
  • A best path of states $\mathrm{S} = \{s^1, \dots, s^T\}$
Do
  • Create a path probability matrix $V$.
  • Foreach $t = 0,1,..,T$
      Foreach $s = 0,1, \dots, N$
          For each transition $i \rightarrow s$
              ▪ $newScore = v[s,t] \times a[s,i] \times b(i, o^t)$
              ▪ if ( $newScore > v[s,t]$ )
                  ▪ $v[i, t+1] = newScore$
                  ▪ $backpointer[i, t+1] = s$
  • Select the highet probability state in final column of $V$
    and backtrack (constract the path $S$)
  • Return S
End
```

Figure 8. The Classification algorithm

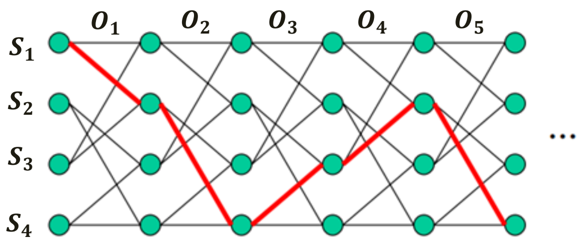


Figure 9. Trellis - a graphical representation of possible state evolution

VI. Experimental Results (Case study)

To evaluate our approach, many experiments were executed. We collected real time datasets from different drilling scenarios. The collected datasets were classified using the classifier described in [9]. The collected datasets and the classification accuracies are described in table 2.

TABLE 2. DRILLING DATASETS

Scenario	#Instances	#Time Series	#Events	Accuracy
#1	18000	10	10	76%
#2	22500	10	9	85%
#3	24000	10	10	95%

The drilling sensors data (time series) used in classification process are described in table 3, and part of these data is shown in Fig. 10.

TABLE 3. STANDARD DATA CHANNELS

Channels	Description
flowinav	Average mud flow-rate
hkldav	Average hook load
mdbit	Measured depth of the bit
mdhole	Measured depth of the hole
posblock	Block position
prespumpav	Average pump pressure
ropav	average rate of penetration
rpmav	Average drill string revolutions
tqav	Average torque
wobav	Average weight on bit

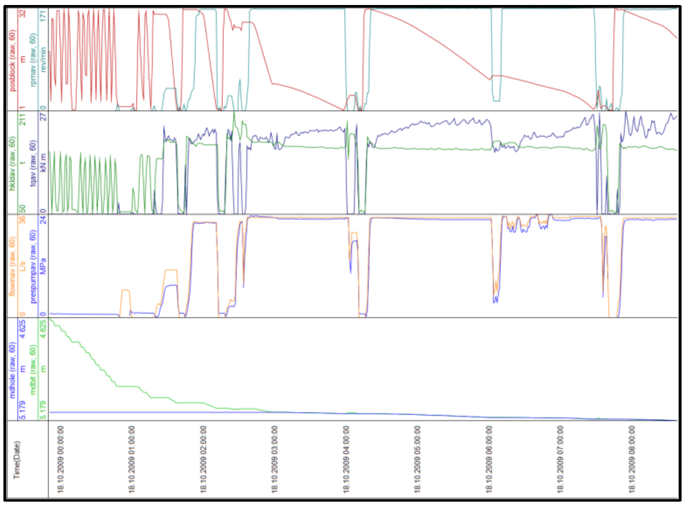


Figure 10. A multivariate time series of drilling data. (Eight variables representing eight mechanical parameters measured at the rig).

To train the model, the confusion matrices were used to calculate the emission matrix as described in Fig. 7. Table 4 shows the confusion matrix of the third scenario. The anonymous names (E1, E2,…, E10) are used to represent 10 different events (classes). As an example, Fig. 11 illustrates how the calculation done for the event E4 which is "CircHL" event.

TABLE 4. CONFUSION MATRIX

	Actual Events										
	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	%
E1	3191	47	236	2	19	1	1	9	0	0	**91.0**
E2	242	3257	502	43	1	12	3	1	0	5	**80.1**
E3	146	147	7328	78	0	2	4	2	0	62	**94.3**
E4	1	2	12	3883	5	20	19	3	47	187	**92.9**
E5	2	0	0	8	525	35	2	5	1	23	**87.3**
E6	0	19	6	60	25	936	11	2	1	82	**81.9**
E7	0	2	2	25	0	26	617	9	30	0	**86.7**
E8	0	0	0	16	9	4	15	317	17	2	**83.4**
E9	0	0	0	6	0	1	8	16	9441	7	**99.6**
E10	0	0	0	1	11	6	1	0	22	10235	**99.6**
%	**89.0**	**93.7**	**90.6**	**94.2**	**88.2**	**89.7**	**90.6**	**87.0**	**98.7**	**96.5**	

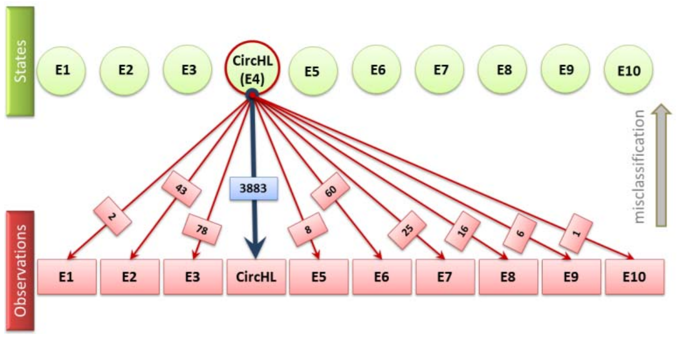


Figure 11. Calculating the emission of the state "CircHL"

In addition to the emission matrix, the transition matrix (state machine) was calculated from the classified data. Fig. 12 shows the transitions from MakeCN (E3) to others states. It is clear that there no transition from E3 event (MakeCN) to E9, E10, E7, E8 and E6 events.

After training a HMM model, this model was used to execute the second cycle classification. Table 5 shows the result. It is obvious that when the classification accuracy is low, the improvement will be significant.

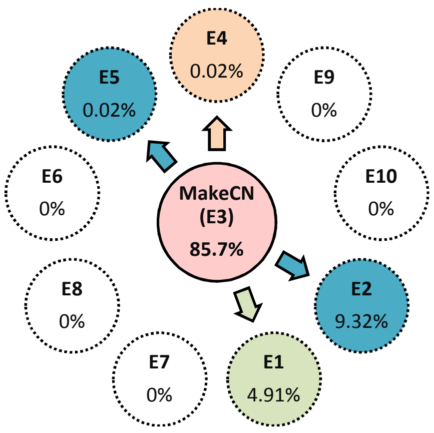


Figure 12. Drilling State Machine (MakeCN)

TABLE. 5 HMM RESULTS

Scenario	First Cycle Classification	Second Cycle Classification	Improvement %
#1	76%	90%	14%
#2	85%	94%	9%
#3	95%	97%	2%

VII. CONCLUSION

The following conclusion can be drawn from the concepts presented in this paper:

- Multiple classifier systems are more accurate than those of single classifier systems.
- Using Hidden Markov Models for sequence classification will improve the classification accuracy of time series classifiers.
- The low classification accuracy we have, the more improvement we get.

ACKNOWLEDGMENT

We thank TDE Thonhauser Data Engineering GmbH for the support of this work and the permission to publish this paper.

REFERENCES

[1] C.A. Ratanamahatana, J. Lin, D. Gunopulos, E. Keogh, M. Vlachos, and G. Das: Data Mining and Knowledge Discovery Handbook 2010. 2nd Edition. Eds. O. Maimon, L. Rokach. Springer. Pages 1049-1077, (2010)

[2] I. Batal, L. Sacchi, R. Bellazzi, M. Hauskrecht, "Multivariate Time Series Classification with Temporal Abstractions". In Proceedings of the Twenty-Second International Florida AI Research Society Conference (FLAIRS 2009), May (2009)

[3] E. Alpaydin, "Introduction to Machine Learning", Second Edition, Massachusetts USA, The MIT Press, February 2010. pp. 351-354

[4] J. Lin, E. Keogh, S. Lonardi, & B. Chiu, "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms". In proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery. San Diego, CA. June 13 (2003)

[5] LR. Rabiner (1989). "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition." Proceedings of the IEEE, 77(2), 257-286.

[6] R. Polikar, "Ensemble Based Systems in Decision Making", IEEE Circuits and systems Magazine, Third Quarter 2006.

[7] Lei Xu, Adam K, Ching Y. Suen, "Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition", IEEE Transactions On Systems, Vol, 22, No3, May/June 1992.

[8] P. Wang, H. Wang, W. Wang, "Finding semantics in time series", Proceedings of the 2011 international conference on Management of data - SIGMOD'11, June 12–16, 2011, Athens, Greece.

[9] B. Esmael, A. Arnaout, R. K. Fruhwirth, and G. Thonhauser. "Multivariate time series classification by combining trend-based and value-based approximations". In Proc. of the 12th International Conference on Computational Science and Its Applications - ICCSA 2012, pages 392–403, 2012.

[10] E. Keogh, K. Chakrabarti, M. Pazzani, & S. Mehrotra, "Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases". In proceedings of ACM SIGMOD Conference on Management of Data. 2011, Santa Barbara, CA, May 21-24. pp 151-162.

[11] J. Hagenauer, P. Hoeher. "A Viterbi algorithm with soft-decision outputs and its applications." In Global Telecommunications Conference, 1989, and Exhibition. Communications Technology for the 1990s and Beyond. GLOBECOM'89., IEEE, pp. 1680-1686. IEEE, 1989.