



MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES
IN PURE AND APPLIED SCIENCES



**Multivariate Time Series Clustering using
Variable Order Markov Models and its
Applications on Cyber-Physical Systems**

BARIŞ GÜN SÜRMEİİ

MASTER THESIS

Department of Computer Science and Engineering

Thesis Supervisor

Assoc. Prof. Dr. Borahan Tümer

ISTANBUL, 2019



MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES
IN PURE AND APPLIED SCIENCES



**Multivariate Time Series Clustering using
Variable Order Markov Models and its
Applications on Cyber-Physical Systems**

BARIŞ GÜN SÜRMEİ
(524115017)

MASTER THESIS

Department of Computer Science and Engineering

Thesis Supervisor

Assoc. Prof. Dr. Borahan Tümer


ISTANBUL, 2019

MARMARA UNIVERSITY
INSTITUTE FOR GRADUATE STUDIES IN
PURE AND APPLIED SCIENCES


Barış Gün SÜRMEĒİ, a Master of Science student of Marmara University Institute for Graduate Studies in Pure and Applied Sciences, defended her thesis entitled “**Multivariate Time Series Clustering using Variable Order Markov Models and its Applications on Cyber-Physical Systems**”, on 19.07.2019 and has been found to be satisfactory by the jury members.

Jury Members


Assoc. Prof. M. Borahan TÜRER (Advisor)

Marmara University, Department of Computer Engineering 

Prof. Çiğdem EROĞĒLU ERDEM (Jury Member)

Marmara University, Department of Computer Engineering 

Prof. Tunga Güngör (Jury Member)

Boğaziçi University, Department of Computer Engineering 

APPROVAL

Marmara University Institute for Graduate Studies in Pure and Applied Sciences Executive Committee approves that Barış Gün SÜRMEĒİ be granted the degree of Master of Science in Department of Computer Engineering, Computer Engineering Program, on 07.08.2019
(Resolution no: 2019/16-02)

Director of the Institute

Prof. Dr. Bülent EKİCİ



ACKNOWLEDGEMENT

I would like to thank European Union's Horizon 2020 research and innovation programme, for funding the IMPROVE research project under grant agreement No. 678867 that I worked in for two and a half years which included a large part of my thesis study.

I would like to thank my advisor Assoc. Prof. Borahan TÜMER for his support throughout my thesis and making it available for me to work in the project IMPROVE and his big effort to make the MInD-NET laboratory available to the students that worked in the project.

Besides my advisor, I would like to thank Assist. Prof. Dr. Peter SCHÜLLER for his scientific support and continuous personal guidance.

I would also like to thank to my colleagues and dear friends in the IMPROVE project, Bilal DİNÇ, Ezdin ASLANCI, Kutalmış COŞKUN and Feyza EKSEN for their support and collaboration.

I also feel the need to thank to my dear friends Ozan OĞUZ, Kemal Toprak UÇAR, Olkan KOÇAK and Sibel KAHRAMAN for their support in my submission process and to Zeynep KUMRALBAŞ for her precious review and feedback.

Last but not the least, I would like to thank to my family, especially my elder brother Salih İlker SÜRMEİ for their priceless support throughout my life.

5.2019

BARIŞ GÜN SÜRMEİ

Table of Contents

1	INTRODUCTION.....	1
2	PRELIMINARIES	5
2.1	Multivariate Time Series Clustering	5
2.2	Markov Models.....	6
2.3	Variable Order Markov Models (VOMMs)	6
2.4	Suffix Trees (STs).....	7
2.5	Probabilistic Suffix Trees (PSTs)	7
2.6	Hidden Markov Models (HMMs).....	9
3	RELATED WORK	11
4	METHODOLOGY.....	14
4.1	Preprocessing - Averaging and Dimensionality Reduction	14
4.2	VOMM MTS Modelling and Comparison.....	15
4.2.1	Discretization.....	15
4.2.2	VOMM Learning	16
4.2.3	VOMM Comparison	25
4.3	HMM MTS Modelling and Comparison	32
4.4	Principal Component Analysis MTS Modelling and Comparison	32
4.5	MTS Clustering	33
5	COMPLEXITY ANALYSIS	35
6	EXPERIMENTAL EVALUATION	37
6.1	Lego Demonstrator Data	37
6.2	Açelik Hydraulic Press Machine Data.....	38
6.3	Setup	39
6.4	Results	40
6.5	Discussion	45
7	CONCLUSION	49
8	REFERENCES	52
	Appendix A. Determining the Number of Clusters in VOMM MTS Clustering	60
	Appendix B. Discretization on a Subset of the Data set and Interpolation on the Rest..	60
	Appendix C. Applications on a Non-CPS Data Set.....	61

ABSTRACT

Multivariate Time Series Clustering using Variable Order Markov Models and its Applications on Cyber-Physical Systems

Keywords: Multivariate Time Series Clustering, Variable Order Markov Models, Cyber-Physical Systems

Multivariate Time Series (MTS) data obtained from Cyber-Physical Systems carry resourceful information about the internal characteristics of the system. As one of the exploratory Machine Learning methods, Multivariate Time Series Clustering can enable one to discover the similarities and differences of the manifested behavior in different working periods/cycles of a system. This information can then be used as a prior knowledge for tasks such as anomaly detection, system maintenance or root-cause analysis. In this thesis, we make use of the statistical method, Variable Order Markov Models (VOMMs) to model each individual MTS and present a new metric to calculate the distances between those VOMMs. The VOMMs are then clustered with respect to these pairwise distances to complete the MTS Clustering task. Two other MTS Clustering methods which use Hidden Markov Models and Principal Component Analysis to model the MTSs are also explained. The superiority of the proposed method is confirmed with the experiments on two data sets; one obtained from a cyber-physical lab demonstrator and one from an industrial dishwasher production plant. A new VOMM construction method as well as the computational complexity of the three MTS Clustering methods are also discussed.

ÖZET

Değişken Dereceli Markov Zincirleri Kullanılarak Çok Değişkenli Zaman Serilerinin Kümeleneşmesi ve Siber-Fiziksel Üretim Sistemlerinde Uygulamaları

Anahtar Kelimeler: Çok-değişkenli Zaman Serileri Kümelemesi, Değişken Dereceli Markov Zincirleri, Siber-Fiziksel Sistemler

Siber-Fiziksel Sistemler'den elde edilen Çok-değişkenli Zaman Serileri (CZS) verisi, sistemin karakteristik özellikleri hakkında değerli bilgiler içermektedir. Bir Makine Öğrenmesi yöntemi olan, Çok-değişkenli Zaman Serileri (CZS) Kümelemesi, sistemin değişik çalışma aralıklarında gösterdiği davranışların arasındaki benzerlikleri açığa çıkarmak için kullanılabilir. Sistem hakkındaki bu bilgiler, hata tespiti, sistem bakımı ve kök neden analizi gibi görevlerin gerçekleştirilmesi için ön bilgi sağlayabilir. Bu tezde, her bir CZS'yi, istatistiksel bir yöntem olan Değişken Dereceli Markov Zincirleri (DDMZ) ile modellenmiş, ve elde edilen bu modelleri karşılaştırarak aralarındaki uzaklıkları/benzerlikleri hesaplamak için kullanılmak üzere yeni bir metrik sunulmuştur. Elde edilen bu ikili uzaklıklar baz alınarak DDMZ'ler kümelendirilmiş ve bu şekilde CZS Kümelemesi görevi sonuçlandırılmıştır. Biri Gizli Markov Modelleri, diğeri ise Temel Bileşenler Analizi kullanarak CZS'leri modelleyen iki yöntem karşılaştırma amacıyla açıklanmıştır. Sunulan yöntemin üstünlüğü, biri siber-fiziksel laboratuvar göstericisinden elde edilmiş, diğeri ise endüstriyel bulaşık makinesi üretim fabrikasından elde edilmiş iki veri seti üzerinde yapılan deneylerle doğrulanmıştır. Ayrıca, yeni bir DDMZ öğrenme yöntemi sunulmuş ve üç CZS Kümeleme yöntemi için hesaplama karmaşıklığı tartışılmıştır.

SYMBOLS

a	: Window size for data averaging
d	: Number of dimensions of a data matrix
k	: Number of data point clusters
mk	: Number of model clusters
t	: Minimum number of occurrences of a sub-sequence to be kept in a Probabilistic Suffix Tree, Pruning parameter
ts	: A set of time series
C	: A set of clusters
I	: Weighting parameter for Enhanced Probabilistic Suffix Tree Matching
L	: Maximum length of any sub-sequence to be kept in a Probabilistic Suffix Tree, Pruning parameter
N	: Number of data points in a data set
W	: Number of iterations for Baum-Welch algorithm
α	: Weighting parameter for Principal Component Analysis Multivariate Time Series Clustering method
μ	: A Set of Multivariate Time Series
μ_{con}	: Concatenated μ
μ_{avg}	: Averaged μ_{avg}
μ_u	: μ_{avg} with reduced dimensionality
σ	: A symbol, one state of a Markov Model
Σ	: Set of symbols, state space of a Markov Model

ABBREVIATIONS

AI	: Artificial Intelligence
CPS	: Cyber-Physical Systems
EPSTM	: Enhanced Probabilistic Suffix Tree Matching
HMM	: Hidden Markov Models
ML	: Machine Learning
MTS	: Multivariate Time Series
MTSC	: Multivariate Time Series Clustering
PCA	: Principal Component Analysis
PST	: Probabilistic Suffix Tree
PSTM	: Probabilistic Suffix Tree Matching
ST	: Suffix Tree
TS	: Time Series
TSC	: Time Series Clustering
VOMM	: Variable Order Markov Models

FIGURES

PAGE

Figure 1: An example visualization of Time Series (TS) data.	6
Figure 2: (top) Suffix Tree of the sequence ABACACABAC\$. Each path from root to a leaf node corresponds to one suffix of the sequence where \$ represents the end of the sequence. (bottom) Probabilistic Suffix Tree of the sequence ABACACABAC (\$ is removed for simplicity) where $t = 2$ and $L = 3$	9
Figure 3: Visualization of subsequence extraction phase (done depth-first by recursion) of VOMM construction procedure applied on the sequence ABACACABAC where $t = 2$ and $L = 3$	19
Figure 4: Visualization of the construction phase (depth-first by recursion) following the subsequence extraction shown in Figure 2.	23
Figure 5: Visualization of the steps of PST Comparison (depth-first by recursion).	28
Figure 6: Steps of VOMM MTS Clustering	31
Figure 7: Lego Demonstrator Photo	37
Figure 8: Demonstration of the setup of the Arcelik Hydraulic Press Machine Data	39

LIST OF TABLES

PAGE

Table 1: The Node List. Sub-sequences and the probability vectors of each node to be inserted in the PST are shown.	20
Table 2: List of all parameters for all methods and their settings used for experiments.	38
Table 3: Adjusted Rand Index Results of PCA MTS Clustering done with the combinations of parameters explained in table 2 on Demonstrator Data. Adjusted RAND Index (ARI) is used for scoring.....	42
Table 4: Adjusted Rand Index Results of PCA MTS Clustering done with the combinations of parameters explained in table 2 on Arcelik Data. Adjusted RAND Index (ARI) is used for scoring.	43
Table 5: Adjusted Rand Index Results of HMM MTS Clustering done with the combinations of parameters explained in table 2 on Demonstrator Data. Adjusted RAND Index (ARI) is used for scoring.....	44
Table 6: Adjusted Rand Index Results of HMM MTS Clustering done with the combinations of parameters explained in table 2 on Demonstrator Data. Adjusted RAND Index (ARI) is used for scoring.....	46
Table 7: Adjusted Rand Index Results of VOMM MTS Clustering done with the combinations of parameters explained in table 2 on Arcelik Data. Adjusted RAND Index (ARI) is used for scoring.	47
Table 8: Adjusted Rand Index Results of VOMM MTS Clustering done with the combinations of parameters explained in table 2 on Demonstrator Data. Adjusted RAND Index (ARI) is used for scoring.	50
Table 9: Adjusted Rand Index Results of VOMM MTS Clustering done with the combinations of parameters explained in table 2 on HASC Data. Adjusted RAND Index (ARI) is used for scoring.	62

Table 10: Adjusted Rand Index Results of PCA MTS Clustering done with the combinations of parameters explained in table 2 on HASC Data. Adjusted RAND Index (ARI) is used for scoring. 63

Table 11: Adjusted Rand Index Results of HMM MTS Clustering done with the combinations of parameters explained in table 2 on HASC Data. Adjusted RAND Index (ARI) is used for scoring. 63



1 INTRODUCTION

Cyber-physical systems (CPSs) are one of the biggest focus of current technological era. They are systems that introduce intelligent components that can learn, solve problems, self-maintain, communicate and help each other. Their ongoing integration in manufacturing systems besides other innovative concepts such as Cloud Computing [1], Virtual Reality [2] and Internet of Things [3] resulted in a new industrial revolution: Industry 4.0 [4]. Referred to as the next generation of automation, CPS are rapidly replacing conventional manufacturing technologies in systems such as production plants, automobile systems or robotic systems.

An application area as CPS with such high impact required the employment of existing Artificial Intelligence (AI) methods, as well as sparking the development of new domain specific methods. Amongst those, Machine Learning (ML) methods promise the automatic analysis and processing of vast amounts of data which are mostly impractical to analyze manually.

One of the most critical issues in manufacturing systems is the everlasting and miscellaneous faults or anomalies which result in high costs and performance degradation in production. Sensor data collected from such systems carry critical information about the system status. Therefore, automatic analysis and processing of data can be a remedy to this problem by detecting, identifying and even finding the root cause of anomalies well before they occur. For this end, ML techniques are employed that can automatically learn from the sensor data the characteristics of the *target system* (i.e., the system under consideration).

In industrial and medical informatics as well as many other fields such as econometrics or weather and earthquake prediction, the collected data consist of data points that are stamped with the temporal information and ordered with respect to time. Most commonly, these time stamps have a constant difference, also called the *sampling period*. This type of data is referred to as *Time Series (TS)*. Time series analysis is a well-studied area in the literature [5]. If the data include only one feature which may for example be the recordings of a temperature sensor, then they are called *Univariate Time Series*. If data contain a collection of features

whose values recorded synchronously, it is referred to as *Multivariate Time Series (MTS)*.

Most of the systems tend to manifest relations between events that occur together or closer in time rather than ones that occur further apart. TS data are therefore crucial for appropriate methods to extract information about temporal patterns of events and causal relationships between those events.

One of the useful data analysis methods is clustering or cluster analysis [6]. The goal of clustering is to find groups of data points with each group manifesting significant similarity within and relative separability outside the group. The similarity of objects is calculated by well-defined metrics. These metrics can be defined: to comprise/include maximum possible features of objects or the relevant domain-specific aspects of the application. It is applied in any area where exploration of such relations is relevant as clustering of images in image processing [7], detecting outliers [8] or extracting information about social groups in social network analysis [9]. In the field of industrial systems this may provide information about internal characteristics of the system behavior.

Time series clustering (TSC) refers to two types of problems in the literature. One is clustering the time series data points which can reveal similar values recorded within specific timespans. In the other problem, time series data recorded in different intervals and referred to as *time series objects* are considered as data points and these are clustered. In CPS, this enables one to identify the time intervals that the behavior of the system is similar at. Extraction of such information from the manufacturing systems can then be utilized for the sake of the critical tasks such as *mode or anomaly detection* [10-12] and *root cause analysis* [13]. In this study we will be dealing with the latter type of these two problems and by TSC we will refer to this problem.

As one might expect, when more than one features is involved the problem is referred to as MTS Clustering. MTS clustering is applied to the data collected from the systems in several fields such as dynamometers [14], earthquake analysis [15], cyber-physical systems [9]. While some of the methods deal directly with raw time series objects [16] some of them work with their higher level abstractions/representations instead which are also referred to as *model-based* methods [17, 18].

One of the preferable methods among powerful mathematical tools to analyze TS data are Markov Models [6]. They can be used to learn the system models that include information of temporal dependencies such as sequential and/or cyclic ordering of events that occur in that system. These dependencies or *significant temporal patterns* can reveal the underlying characteristics and behavior of the target system that TS data are obtained from. In Markov Models, a system is assumed to be at one of a finite number of possible states during a specific time step. In practice, this time step can be as short as or longer than the sampling period of the data obtained from the system. A type of Markov Models is Hidden Markov Models (HMMs). In HMMs the observations do not explicitly correspond to states and the states are hidden, but the observations are assumed to provide information that can be used to estimate the hidden states. Both regular Markov Models and HMMs assume dependencies of identical context length. This is insufficient for analyzing a target system where dependencies with different context lengths may exist. A common such domain is natural language processing where one has to study words with different lengths. Variable order Markov models (VOMMs) [19] can be a much better choice to tackle this problem, which can represent temporal dependencies of *variably long patterns* by allowing dependencies between the events that occur within variably many time steps.

As to the novelty of this work, the contributions are six-fold and listed in the following:

- We propose an MTS Clustering method which
 1. learns the significant patterns that are manifested in MTS as VOMMs,
 2. compares them, and,
 3. clusters them regarding the results of this comparison.
- We propose a VOMM construction algorithm, which is easy to understand and implement while having comparable complexity to the algorithms proposed in the literature.
- We propose
 1. an improved version of our VOMM comparison formula PSTM [10], and,
 2. a linear time comparison algorithm which realizes the proposed formula,

- We compare PSTM to a recently proposed Frobenius Intersection VOMM comparison method,
- We analyze, test and evaluate our MTS Clustering method with two different methods proposed in the literature, one PCA-based [17] and one HMM-based [18] MTS Clustering and confirm the superiority of our method on two industrial datasets, (1) semi-physical Lego Lab Demonstrator Data and (2) real world Arçelik [20] Hydraulic Press Machine Data, and finally
- We discuss the time complexity of the proposed methods and show that the proposed method has comparable performance regarding performance while having better clustering accuracy.

In Section 2, we summarize the MTS Clustering Problem, VOMMs as well as their realization method PSTs and HMMs. Related work is discussed in Section 3. In Section 4 we discuss how each MTS model is learned, how distances between these models are calculated, and how we perform clustering using the calculated distances. In Section 5 we give an extensive complexity analysis of the presented methods. Section 6 contains our experiments and a discussion of the results. We conclude the paper in Section 7.

2 PRELIMINARIES

2.1 Multivariate Time Series Clustering

Time series (TS) data include time information so that points are ordered with respect to time and the corresponding time label for each consecutive point couple have a constant difference. If $t = \{t_1, t_2, t_3, \dots, t_N\}$ are time steps t_n that subsequently increase by a constant, then a time series can be written as $ts = \{ts_1, ts_2, ts_3 \dots ts_N\}$ where each element ts_n corresponds to a time unit in t_n .

If points in data have more than one dimension it is referred to as multivariate data. In multivariate data, each data point is a d dimensional vector. Accordingly, if time series data have multivariate data points, it is referred to as Multivariate Time Series. Then an MTS can be written as $M = \{m_1, m_2 \dots m_N\}$ where each element $m_n = [x_1 \ x_2 \ x_3 \ \dots \ x_d]$, a d dimensional vector, corresponds to a time unit in t_n .

Given a set of MTSs $\mu = \{M_1 \ M_2 \dots M_X\}$ where each M_x has a number of d -dimensional data points and a set of clusters $C = \{c_1, c_2 \dots c_L\}$, the *MTS Clustering* problem can be defined as finding a mapping from each M_x in μ to a c_l , where each c_l holds M_x s displaying a *significant* similarity. An example scenario where this problem applies is visualized in Figure 1.

Clustering is applied either to MTSs or their corresponding higher-level abstractions i.e. models, either way a similarity or distance metric should be defined to calculate the distances. In this study we limit our scope so that each MTS is considered to have exactly d -dimensional data points.

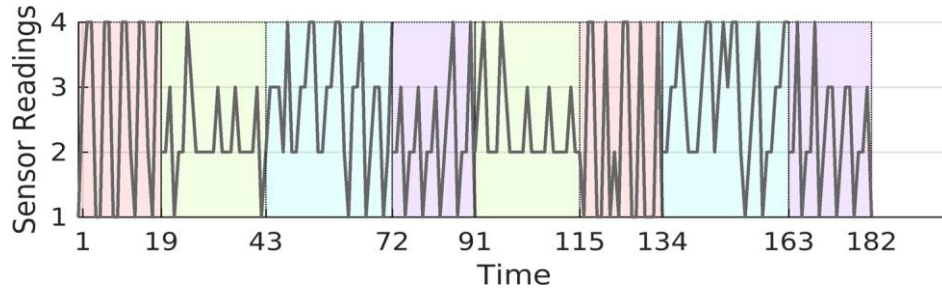


Figure 1: An example visualization of Time Series (TS) data. Such data can be obtained by recording the value in equally spaced time steps that a sensor measures which is placed in a Cyber-Physical System. One can conclude that the system behaves differently in different periods of time and those different types of behavior manifest themselves in the data by different patterns (as separated by lines and shown with different colors in the figure). Considering each of the periods as different TS objects, clustering techniques can be applied to find out different groups of periods, such that in each group, only the TSs that correspond to a certain behavior exist.

2.2 Markov Models

Stochastic processes are processes that explain the change of random variable(s) in time. They are used to represent real world system processes (industrial, biological, chemical etc.) where values of those variables for a specific time step is considered to explain the *state* of the system at that moment. Markov property is defined as a property of a stochastic process so that system's next state being dependent only on the current state of the system and independent of its previous states [21]. Markov Models are probabilistic models that have Markov property. Such a property of a model enables tractability and feasibility of its estimation. An extension to this definition is the n^{th} -order Markov Models where the next state is dependent on the n most recent states of the system before the current state. If the state space of the modeled system is given by the symbols of the alphabet $\Sigma = \{S_1, S_2, \dots, S_k\}$ then n^{th} -order Markov Model is defined by two components, (1) the initial state probability vector $\overline{P(S)}$ where $S \in \Sigma$ and (2) the transition matrix that contains the probabilities of the system going to a state following the last n states the system has been at, or the *context*, $P(S_{t+1}/S_{t-n+1}S_{t-n+2} \dots S_t)$.

2.3 Variable Order Markov Models (VOMMs)

Another extension to Markov Model definition is Variable Order Markov Models [19]. Unlike n^{th} order Markov Models which contain only probabilities of state transitions from

the state sequences with context length n , VOMMs contain the probability information for state sequences of different length/order with a maximum length of L . Rather than keeping information for all the possible state sequences for permutations of the symbols of Σ and all possible lengths up to L , the information for specific sequences are kept in the model depending on the *significance* of the sequence regarding the data. This significance is determined by observing the occurrence frequency of the sequence in the data. The higher the occurrence frequency, the more probable that the sequence will be stored in the model. This property yields several advantages over the Markov models with a fixed order such as *tractability*; where the model can be estimated by employing a reasonable amount of computational resources, *adaptivity*; where it can be used to represent the behavior of real-world processes such as the processes in an industrial plant or frequently occurring patterns in biological structures such as in protein sequences *expressivity*; the model can extract information about more diverse types of behavior, lower computational complexity; where the model is much more efficient by means of space and time compared to overall complexity of the fixed order Markov models from 1st, 2nd, ..., L^{th} order combined.

2.4 Suffix Trees (STs)

Suffix trees are representations of any sequence which provide minimum-cost (i.e., by a minimum number of operations) accessibility to any sub-sequence of that sequence by traversing the tree from its root to leaves. By concatenating the subsequences on the edges for traversal of each path from root node to a specific leaf node yields a unique suffix of the sequence. An internal/non-leaf node exists if and only if more than one suffix shares a subsequence and this subsequence is kept in that internal node (see Fig. 1 (top)). Depending on the purpose of use, nodes may be used to keep the concatenation of the subsequences that are on the edges in the path from the root to that node.

2.5 Probabilistic Suffix Trees (PSTs)

Probabilistic Suffix Trees [22] are one way to realize VOMMs. They are related to Suffix Trees and have similar structure as they can be constructed upon STs as a probabilistic variation [23] or directly as PSTs [22]. Different formulations of PST exist in the literature [24, 25] as explained in [23] and in this thesis, following explanation is based on the

formulation in [25]. A PST of a sequence contain probabilistic information about the subsequences that represent the characteristics of the sequence in a concise way. Each node contains a subsequence and a probability vector that keeps the probabilities of the occurrence of each symbol in the alphabet following that subsequence. Subsequences that occur in the sequence will not be in the PST if; (1) it does not appear in the modeled sequence more than a minimum occurrence parameter t (t pruning) or (2) the subsequence is longer than the maximum length L (L pruning). In other words, t prunes the subsequences that does not occur significantly many times and the detail of the model is limited by L . The probability information is added to the nodes in the form of a probability vector that keeps the probabilities of each symbol/state occurring after the corresponding subsequence/state sequence of the node. An example sequence and its constructed PST is shown in Fig. 1 (bottom). One should notice the difference in the ordering of nodes which have parent-children relation in STs and PSTs: the subsequences of the nodes are extending from the beginning of the subsequence in PSTs ($A \rightarrow BA$) and vice versa in STs ($A \rightarrow AB$). This is so since n th level in the PST corresponds to n th order in Markov models and this structure enable *optimal complexity* for subsequence search in the tree for prediction. For example, if $P(A/AB)$ is requested, *longest suffix* of the given subsequence AB will be searched in the tree as it is the *most relevant* information. In the case of the PST in Figure 1, AB is in the tree and the probability $P(B/CA) = 0.5$ will be obtained. If $P(B/AA)$ is requested, CB is not in the tree and the model will use the most relevant information as a prediction: $P(B|A) = 0.4$. Therefore, for $P(A/AB)$, this structure of the PST allows one to access the node of B and then access that of AB if it exists in the PST, which is done in $O(m)$ time given the length of the subsequence is m .

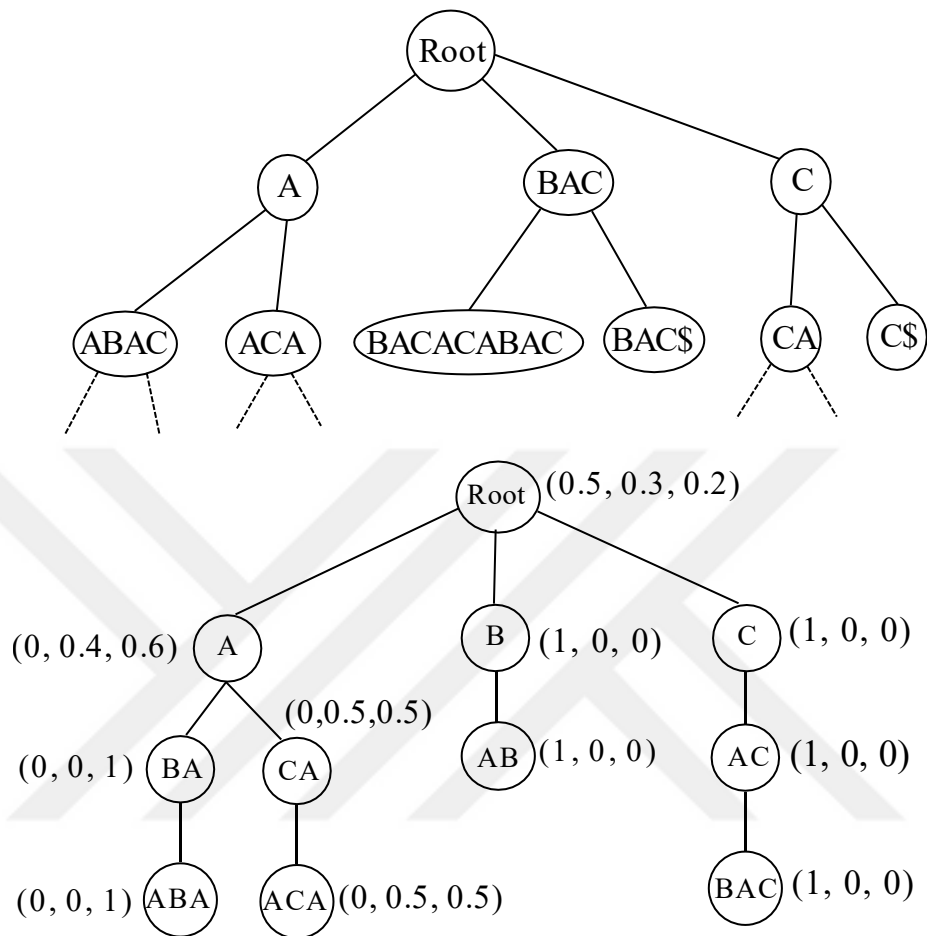


Figure 2: (top) Suffix Tree of the sequence ABACACABAC\$. Each path from root to a leaf node corresponds to one suffix of the sequence where \$ represents the end of the sequence. (bottom) Probabilistic Suffix Tree of the sequence ABACACABAC (\$ is removed for simplicity) where $t = 2$ and $L = 3$.

2.6 Hidden Markov Models (HMMs)

In Hidden Markov Models, the state sequence and the state space are assumed to be non-observable. Observable data are called *Observations* and the goal is to estimate the underlying behavior of the system for the hidden states via processing the information extracted from these observations. If the set of hidden states is Q and set of observations are O , HMM is represented by three parameters: (1) initial state probability vector Π , (2) transition matrix A which holds the transition probabilities from each state to any other state and (3) emission matrix B that consists of the probabilities for each hidden state that any observation is to be observed while the system is actually in that state. Most common way of

learning these parameters is the Baum-Welch algorithm [26] which is an Expectation-Maximization algorithm. Baum-Welch algorithm makes use of a forward-backward procedure.



3 RELATED WORK

There is extensive work done in data analysis literature on Time Series Analysis [27]. Application fields include but not limited to; health informatics [28, 29], cyber-physical systems [5] and bioinformatics [30-32], seismology, meteorology, econometrics as well as signal detection and estimation.

The analysis generally aims process monitoring, tracking business metrics or forecasting the values of a variable or predicting the behavior of the analyzed system. Technically, methods try to extract the patterns manifested in the data over time which may be disguised by noise.

First approaches to time series analysis started with visually observing the simple records gathered by tracking the features/variables regarding specific processes, such as the number of an item sold per day over a time period. The analyzers tried to relate the specific changes of the features and the environmental incidents occur concurrently. Technological evolutions resulted in both appearance of large-scale systems that require more complex analysis and possibility to record time series data with much higher frequency and including many features at the same time. This led the analyzers to use machine based, automated systems as well as integrating complex mathematical models.

Conventional methods of time series analysis include modelling approaches where the goal is to extract meaningful statistics or patterns from the data by explaining it with specific mathematical functions such as auto-correlation function or spectral density function [33]. These methods are generally applied in areas such as signal processing, control engineering and electronics. They are classified into two, Frequency Domain Methods which include spectral and wavelet analysis and Time Domain methods which include auto-correlation analysis.

Machine Learning (ML) methods are also employed to learn the models of the time series data with less or without explicit instructions compared to conventional methods. These include model learning techniques such as Artificial Neural Networks, Support Vector Machines or Markov Techniques [6].

ML techniques are extensively used for forecasting. A review of such techniques is given in

[34] which classified these techniques into global and local learning techniques. In TS forecasting task, Local learning approach is preferable as it does not assume the underlying system that emits the TS data to maintain a specific behavior all the time. Instead, it focuses on local patterns that may be manifested in the data and tries to extract them independent from each other. VOMMs also enable local learning by extracting cycles/order of sequences, namely context, occur in different time intervals in the TS data.

A comprehensive work on the methods that has been proposed on TS Clustering problem is presented in [35]. It includes categorizations of the methods by many aspects such as:

- Representation method, namely if the raw time series are directly compared or their higher-level abstractions/models are constructed in prior and these are compared,
- Comparison method, raw time series distance calculation method or if the models of the TS are constructed, then comparison technique of the models,
- Clustering method which is used once the distances between TSs are calculated.

Collection of the methods in the following ten years are surveyed in [36]. Presented methods mainly applicable for Univariate Time Series which includes the well-known successful method SAX [37], number of methods that can deal with Multivariate Time Series are relatively low.

An important distinction between the existing MTS Analysis methods is that while most directly works on multivariate, possibly continuous data, some uses *discretization* or *symbolization* methods [38] that transforms them to univariate data and limits the data space to a finite set of symbols. Discretization is generally followed by employing appropriate model learning methods, for instance statistical methods like Markov Models [39]. Various ways to do the discretization task are presented in the literature; Probability Density Function partitioning [37], employing decision trees to divide the data space [40], via clustering the data points [40], using self-organizing maps [41, 42] or mining temporal-interval relation patterns [43].

Methods for anomaly detection including Markov model-based representations of data are surveyed in [11].

VOMMs are applied in many areas for classification and prediction tasks [22, 44, 45]. As a common way of realization, Probabilistic Suffix Trees (PSTs) [24, 25]. [23, 46] showed that it is possible to construct PSTs in linear time. Application areas of PSTs include but not limited to bioinformatics [22, 23, 44, 46], outlier detection [47] and cyber-physical systems. The applications in bioinformatics contain protein family prediction, where the state sequence alphabet is the types of amino acids and the models of those sequences are learned as PSTs. Then, by running new protein sequences on those learned models which correspond to each of the family, the probabilities of the sequences falling into one of the families are calculated. In the same way, PSTs are also used for DNA binding site classification. Another area that VOMMs are extensively applied is lossless text compression [48-50]. Thanks to their concise structure, VOMMs enable such compression with their ability to not keep redundant prefixes of the substrings in the text as in Suffix Trees. Another interesting field of use is music generation [51], where the VOMM is trained with a data set of songs with specific music style. Then the learned conditional probabilities of the model are used so that the most likely state sequences according to the model are combined to generate music of that style. VOMMs were specifically found successful of capturing the stylistic information. Another Markovian Technique, Hidden Markov Models [6] are also applied in similar areas as in bioinformatics [52]. A comparison of VOMMs and HMMs is given in [22] and superiority of VOMMs over HMMs with respect to performance while having comparable prediction power is shown. Other unsupervised methods in cyber-physical systems employed Probabilistic Automata, Bayesian Networks and Self-Organizing Maps [53, 54, 42].

Unlike the method proposed in this thesis, state-of-the-art methods for classification of input sequences employ PSTs, simulate a run of the test sequences on PST models learned in a training phase to decide which class the sequences belong to. A method that was introduced in [55] used (non-probabilistic) STs by representing them as a vector space document model and compared them for clustering.

4 METHODOLOGY

Here we will describe three different approaches to solve the MTS Clustering (MTSC) problem. All methods follow three main steps: (1) model learning, (2) comparison and (3) clustering where in (1) the characteristic information of the MTS is learned as a model, in (2) the learned models are compared to obtain a dissimilarity matrix that contains pairwise distances between models, and in (3) MTS are clustered using the dissimilarity matrix.

In this section, we explain three different methods that deal with MTS Clustering (MTSC) task. All methods share three subtasks where one's output is the input of the next: first, the construction of higher-level abstractions/models of the MTSs. Second, the comparison of all constructed models which yields a dissimilarity matrix and third, clustering the MTSs using the dissimilarity matrix.

In sections 4.2 through 4.4, we describe the model learning and comparison steps for each of the 3 clustering methods. Since we apply the same preprocessing step for all 3 methods and clustering techniques on the dissimilarity matrices, we discuss them in sections 4.1 and 4.5, respectively.

In section 4.1 we describe the common data preprocessing techniques applied in all three methods. Sections 4.2, 4.3 and 4.4 clarify the unique parts of the methods which consist of the model construction and the comparison of those models. In section 4.5 we discuss the common clustering techniques that are applied commonly for all three methods.

4.1 Preprocessing - Averaging and Dimensionality Reduction

One of the characteristics of time series data obtained from cyber-physical systems is that the sampling rate is usually so high that changes in the behavior of the system are rarely observed. Another effect of the high sampling rate is that instantaneous distortions in the sensor values may occur. Therefore, without an appropriate preprocessing, the parts of the data with useful information might be masked or dominated by noise fluctuations or by the large intervals where there is no change. To avoid this, in this study we apply *averaging* on the time series. Consider that M_{con} with size $N \times d$ is the concatenation of all MTSs and it is then divided into windows which contain a data points. Then each window is a matrix with

size $a \times d$. Averaging is done so that each window is replaced with a d dimensional data point which, for each dimension, consists of an average of a values in the window for that dimension. Averaged dataset is referred to as M_{avg} which has the size $N/a \times d$.

Industrial data may contain an extraordinarily high number of variables/features which may require a very high amount of computational power to process. As an attempt to deal with this and reduce the noise, PCA is applied and d dimensional M_{avg} is transformed to a u dimensional M_u where $u < d$ in VOMM and HMM MTS Clustering methods. This is not done for PCA MTS clustering since it already applies PCA in its model construction step. Therefore, following steps are applied on M_u for VOMM and HMM Clustering and on M_{avg} for PCA Clustering.

4.2 VOMM MTS Modelling and Comparison

In this section, we present the steps of (1) model construction and (2) comparison of the method proposed for VOMM MTS Clustering. VOMMs are not directly constructed upon the preprocessed MTSs (M_u), they are first transformed to a one-dimensional, categorical/symbolic sequences. This is done by mapping each data point in M_u to a symbol σ of a finite alphabet Σ . To achieve this, all data points in the dataset are clustered regardless of their time information, as they were a static dataset instead of time series data. For each cluster, data points in that cluster are labeled with a unique symbol σ . Replacing the data points with these labels, the MTSs M_x in μ (see 2.1) are reconstructed as sequences p_x which forms the set of sequences P . For each sequence p_x in the set of sequences P which corresponds to the MTS M_x in μ , a separate VOMM ϕ_x is constructed to form the set of VOMMs Φ . these VOMMs are compared pairwise with respect to distance metrics which will be explained in section 4.2.3.

4.2.1 Discretization

Discretization is accomplished by clustering applied on M_u where each of the l clusters are labeled with a unique symbol σ from a finite alphabet Σ with size l . Accordingly, by replacing the data points in M_u with the label of the cluster they fall in, a *discrete* sequence is obtained. We refer to this task as *data point clustering* throughout this paper to be able to distinguish

this from the other clustering task we perform in the final step of our MTS Clustering method to group the learned VOMMs.

To discretize the data set, clustering is applied on M_u . Each one of the l clusters are labeled with a unique symbol σ of a finite alphabet Σ with size l . According to this labeling, for each MTS, data points in that MTS are replaced by their corresponding label. As a result, for each MTS, a discrete, univariate sequence is constructed from tokens. In order to distinguish the clustering applied here from the clustering that is applied when clustering the VOMMs (see section 4.5), we refer this one here as data point clustering throughout this paper.

Data sets stemming from industrial systems typically contain a large amount of data points. Therefore, it is challenging to obtain stable results in a reasonable amount of time. We use Ward’s Hierarchical Agglomerative Clustering [56], a deterministic method with acceptable time complexity of $O(n^2)$ where n is the number of data points.

4.2.2 VOMM Learning

For each of the discrete sequences p_x obtained corresponding to MTS m_x , a VOMM is learned so that each of M_x in μ (see 2.1) corresponds to a VOMM ϕ_x of a set Φ of VOMM’s. VOMMs are learned as PSTs.

In [23] a PST construction algorithm referred to as AV-2 is proposed which constructs PSTs of sequences by first obtaining the ST and then pruning the nodes of the ST, adding the probability vectors and then adding the *reverse suffix links* (*rsuf’s* introduced in [46]). The *rsuf’s* allow one to access the probabilities $P(\sigma_3/\sigma_2)$ and $P(\sigma_3/\sigma_2\sigma_1)$ consequently, and therefore enable the optimal subsequence search explained in section 2.5. The STs are constructed by the method *lazy suffix tree* proposed in [57] which, while having $O(n^2)$ worst-case complexity, in practice, is shown to outperform other ST construction methods which have linear worst-case complexity such as the well-known theoretically optimal algorithm proposed by Ukkonen in [58]. Likewise, in [23], AV-2 is experimentally shown to outperform the theoretically linear time PST construction algorithm proposed in [46]. It is also argued in [23], the linear time algorithm proposed in [46] is quite complex. However, one of the complexities of the approach is the addition of *rsuf’s* which is also used in [23].

We propose a PST construction algorithm which employs the sequence traversal technique used in *lazy suffix tree* algorithm, but unlike *AV-2*, PST construction does not follow a ST construction; instead they are constructed directly in the structure explained in section 2.5 as explained below. Therefore, while having the same complexity properties, we believe that our method is simpler to understand and implement.

PST Construction is accomplished in two main phases. In the first phase which we call *subsequence extraction*, for each one of the subsequences that are eligible to be kept in the model, a node object is initiated that keeps the subsequence. Then the next symbol probability vector for the subsequence is calculated and kept in the initiated node which is added to a “node list.” In the second phase which we call a *tree construction*, PST is constructed by inserting the nodes as in the structure explained in section 2.5.

Concretely, PST Construction from a sequence starts with initiating a “suffix index list” (SIL) that keeps the indices to the *first characters* of all suffixes of the sequence. If SIL is implemented as an array, these indices are enough to represent and access to the suffixes of the sequence in $O(1)$ time.

The subsequence extraction phase of PST construction algorithm consists of three main steps: (1) Find the *longest prefix* of the suffixes represented in the SIL and increment all the indexes in the SIL by the length of the longest prefix, (2) initiate nodes with all the *prefixes of the longest prefix* and add them a list called “node list”, (3) sort and group the elements of SIL with respect to the characters that incremented indexes in SIL. Recursively, for each group new SILs are initiated and same three steps are applied. Notice that by incrementing the indexes in step (1) the represented suffixes change accordingly. The indexes that are larger than the length of the sequence or L (see 2.5) are immediately removed from the list they belong to. No list is created, nor the steps are applied for the groups with less than t elements (see 2.5). Subsequence extraction phase is visualized in figure 2.

Tree construction phase is similar to subsequence extraction phase. Instead of the SIL, following steps are accomplished recursively with the node list obtained in subsequence extraction phase: (1) Find the node with the shortest subsequence, S , in node list. (2) Insert S to the tree and remove it from node list. (3) Counting sort and group the node list according

to the n^{th} last element of their subsequence where n is initially 1 and incremented as the processed depth level of the tree at that moment increases. Recursively for each group, new “node list”s are initiated and same steps are applied. Note that for any node list that contains the nodes with the subsequences in the form of ΔS , there is only one node with the *shortest* subsequence S that will be the root of the subtree that will be built with that node list. Tree construction phase is visualized in figure 3.

As in [57], sorting the suffixes are done using Counting Sort [59] which has worst-case complexity $O(n + k)$ where n is the number of elements to be sorted and k is the range of distinct keys to be sorted. Therefore, it has linear complexity if k is not significantly greater than n . In the case of sorting suffixes, keys are the elements of the symbol alphabet Σ (see 2.4). In the context of MTSC using VOMMs, the size of Σ is expected to be much smaller than the size of the modeled sequences and eventually number of their suffixes and therefore Counting Sort takes linear time and appropriate to use.

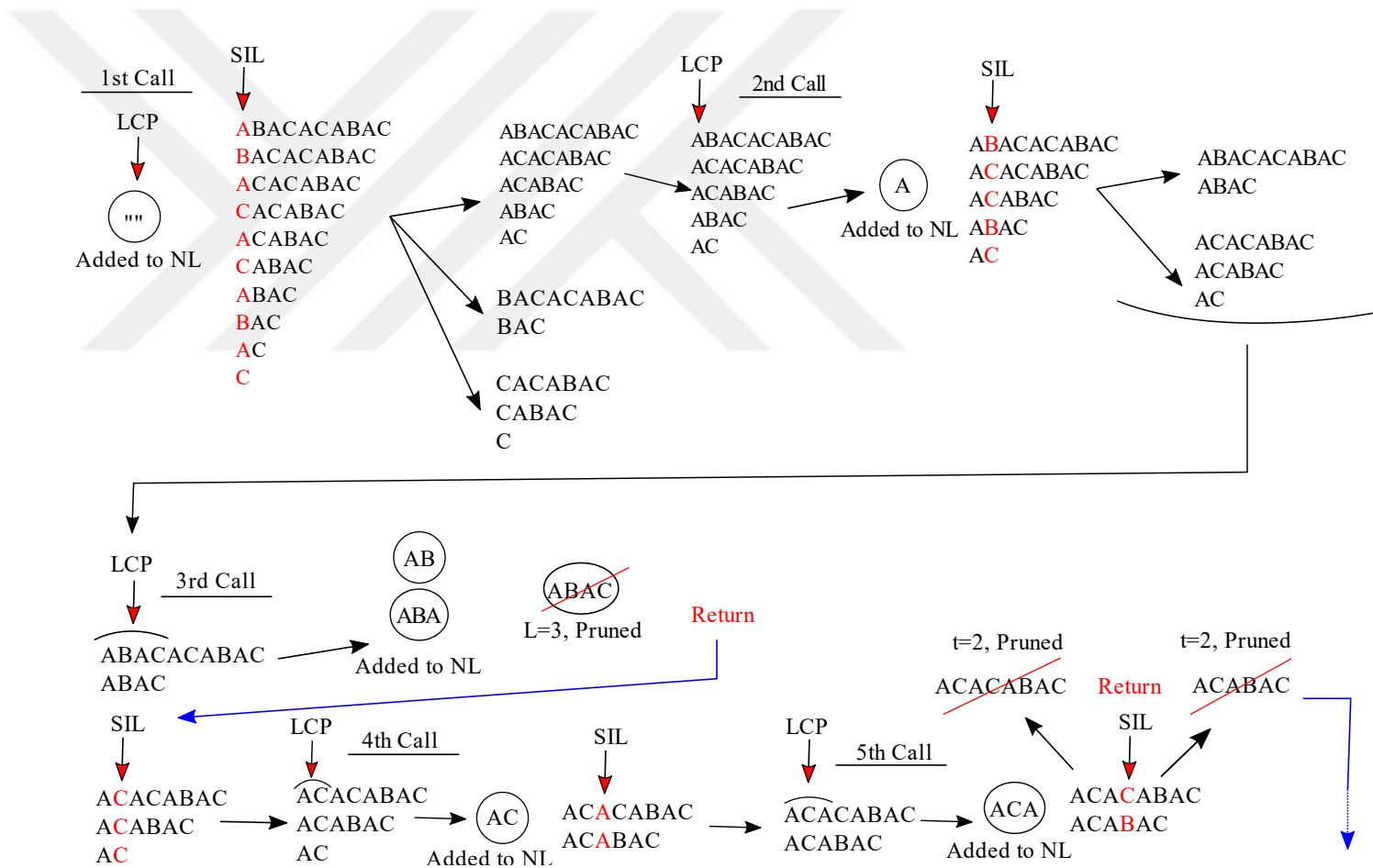


Figure 3: Visualization of subsequence extraction phase (done depth-first by recursion) of VOMM construction procedure applied on the sequence ABACACABAC where $t = 2$ and $L = 3$. Initially, indexes in SIL correspond to the first characters of the suffixes of the sequence. Then in each step/call, nodes with the prefixes of the longest common prefix (LCP) and their probability vectors are added to the Node List. Then the indexes in SIL are grouped with respect to the characters corresponding to them in the sequence. Grouped indexes are incremented to the characters that follow the LCP and new SILs are created with each of those groups. In the following steps, these procedure is applied to the new SILs. Return of each call of the recursion is shown with the blue arrows.

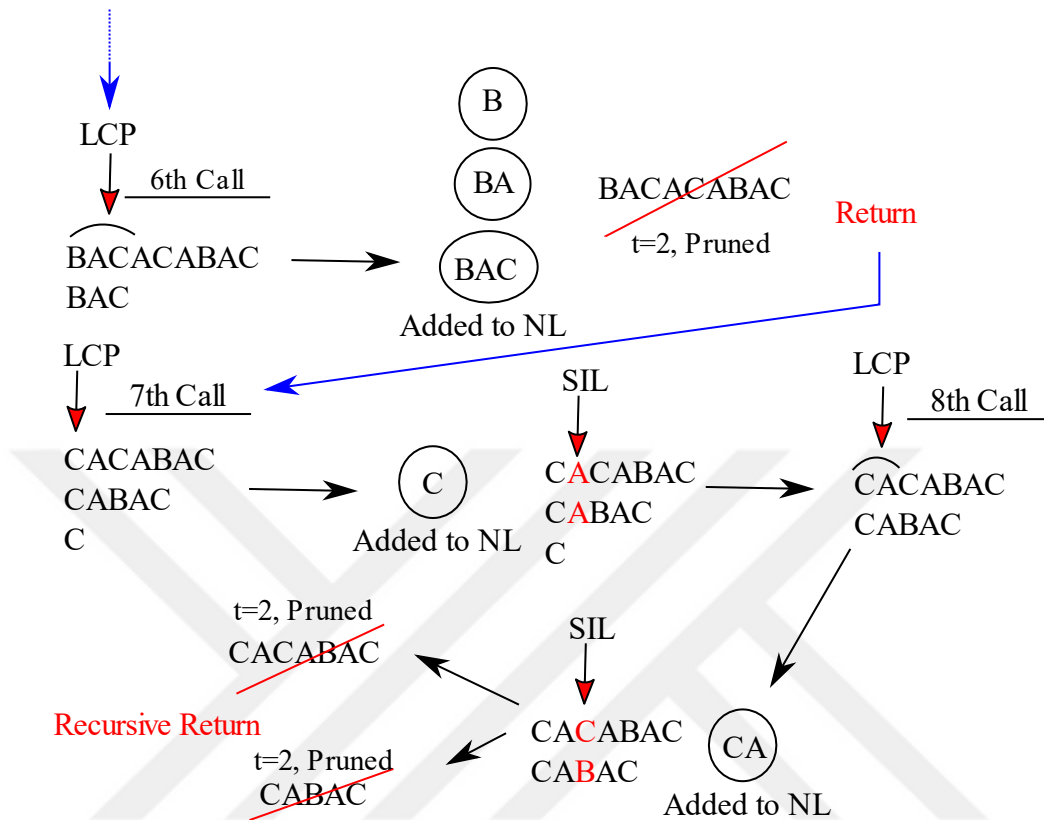


Figure 2 Continued

Table 1: The Node List. Sub-sequences and the probability vectors of each node to be inserted in the PST are shown.

Added in Call	Sub-Sequence	Probability Vector
1st	""	(0.5, 0.3, 0.2)
2nd	A	(0, 0.4, 0.6)
3rd	AB	(1, 0, 0)
3rd	ABA	(0, 0, 1)
4th	AC	(1, 0, 0)
5th	ACA	(0, 0.5, 0.5)
6th	B	(1, 0, 0)
6th	BA	(0, 0, 1)
6th	BAC	(1, 0, 0)
7th	C	(1, 0, 0)
8th	CA	(0,0.5,0.5)

A detailed pseudocode of the PST Construction Algorithm is given below.

Class Node:

1. Initialize(this, Subsequence)
 1. this.Sbsq := Subsequence
 2. this.Probs := [] // Probability vector
 3. this.Children := an array of Null pointers with the size of the Alphabet

CONST_PST (Sequence, Alphabet)

1. SIL := Suffix Index List, indexes of all suffixes in the Sequence
2. Global Var. Node List := []
3. Global Var. seq := Sequence
4. GET_NODES (SIL)
5. Global Var. alph := Alphabet
6. PST Root := BUILD_TREE (Null, Node List, 0, -1)
7. Return

GET_NODES (SIL)

1. Find the length of longest common prefix, LLCPC
2. BrnchNode := GET_PREFIX_NODES(SIL[any element], LLCPC)
3. Increment all the elements of SIL by LLCPC, remove anyone reach size(Sequence)
4. If LLCPC <= L:
 1. Counting sort the SIL wrt. Characters in Sequence indexed by elements of SIL
 2. Initiate a “Group SIL” that keeps for each of σ in alph, a list of indexes which satisfies $\sigma = \text{seq}[\text{SIL}]$
 3. For i from 1 to size(alph): // set the probability vector for current node
 1. BrnchNode.Probs[i] := size(Group SIL[i]) / size(SIL)
 4. For i from 1 to size(alph):
 1. If size(Group SIL[i]) >= t // t-Pruning

1. GET_NODES (Group SIL[i])

5. Return

GET_PREFIX_NODES(First, LLCP)

// initiate nodes with the prefixes of longest common prefix and add them to Node List

1. For i from 1 to LLCP:

1. If $i \leq L$: // L-Pruning

1. Current Node = Node (Sequence[First : First + i])

2. If $i \neq LLCP$:

1. Set Current Node.Probs so that the corresponding probability value for Sequence[First + i + 1] is 1 and other values are 0

3. Else:

1. Branching Node := Current Node

2. Else:

1. Break

3. Add Current Node to the Node List

2. Return Branching Node

BUILD_TREE (Root, Node List, CT, childIndex)

1. S = Node in the Node List with the shortest subsequence

2. If childIndex $\neq -1$, Root.Children[childIndex] := S

3. Remove S from Group Node List

4. CT := CT + 1

5. Sort (Counting sort) the nodes in the Node List wrt. Node.Sbsq[size(Node.Sbsq) - CT]

6. For each character σ in the alph initiate a "Group Node List" which keeps the nodes that satisfy $\sigma = \text{Node.Sbsq}[\text{size}(\text{Node.Sbsq}) - \text{CT}]$

7. For i from 1 to size(alph):

1. BUILD_TREE (S, Group Node List, CT, i)

8. Return S

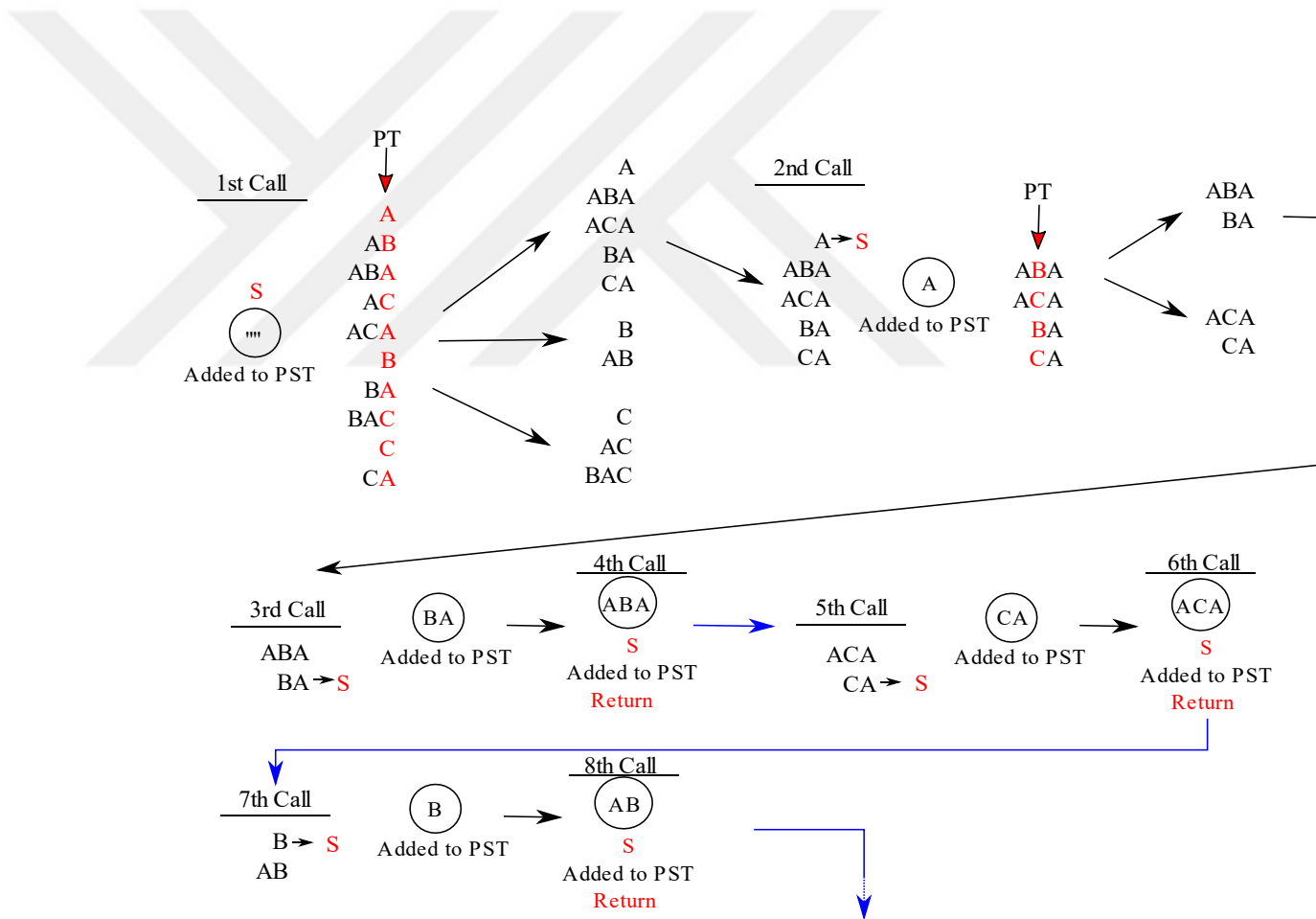


Figure 4: Visualization of the construction phase (depth-first by recursion) following the sub-sequence extraction shown in Figure 2. In each step/call, node with the shortest subsequence is inserted to the tree as a child of the node inserted in the previous level of the currently processed branch of the tree. Remaining nodes are grouped according to the n^{th} last character of their sub-sequence (shown as PT) where n is initially 1 and incremented as processed depth level of the tree at that moment increases. New Node Lists are created with those groups and the same procedure is applied in the following steps to those Node Lists.

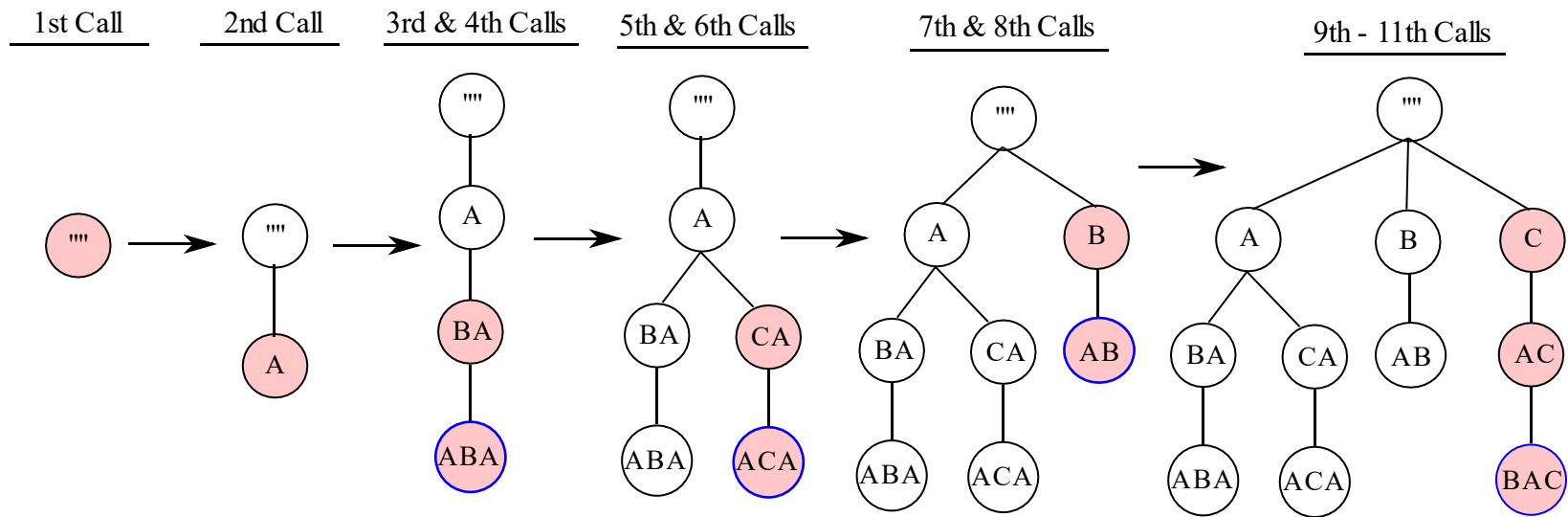
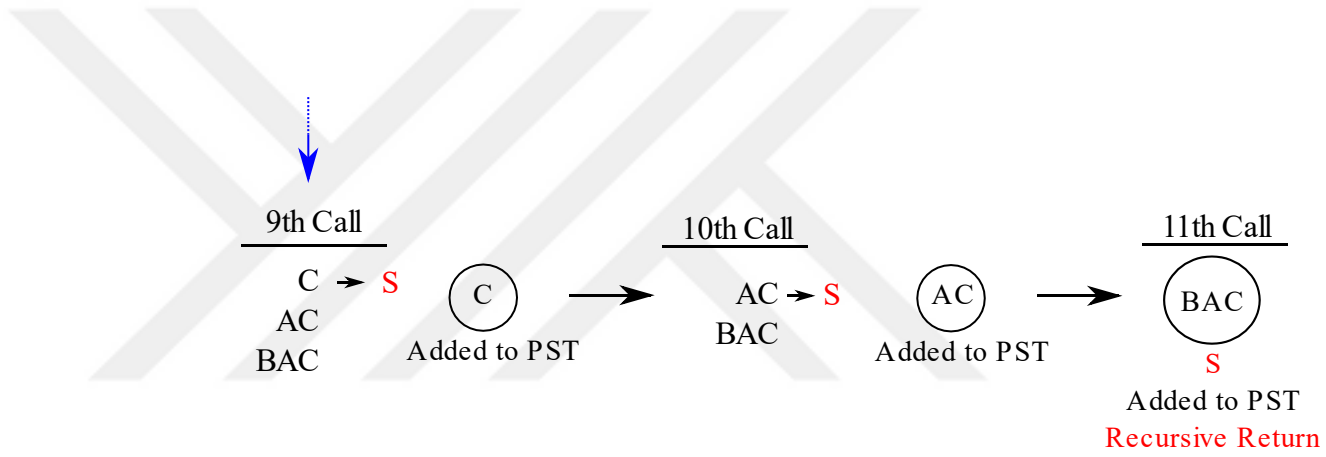


Figure 3 Continued

4.2.3 VOMM Comparison

To obtain a dissimilarity matrix that contains the pairwise distances of all VOMMs in Φ , a distance metric should be formalized for comparing any pair of VOMMs. We use two different VOMM comparison methods: we propose Enhanced PST Matching which is an improved version of PST Matching (PSTM) [10] and compare it to Frobenius Intersection proposed in [60].

Enhanced PST Matching: We start shortly explaining PSTM method [10].

PSTM calculates the distance between two PSTs by two aspects, (1) *Probability Distance*, which calculates the differences of the values of the same model parameter, namely probabilities of the occurrence of same type of observation and (2) *Structural Distance*, calculates the structural differences between two trees which actually occur when a model parameter/node exists in one model/tree and not in the other.

Consider two sequences A and B and PSTs constructed from these sequences T_1 and T_2 respectively. We define two sets, $\{A_1, A_2, \dots, A_N\}$ and $\{B_1, B_2, \dots, B_M\}$, respectively contain the subsequences that exist in nodes $\{K_1, K_2, \dots, K_N\}$, and $\{R_1, R_2, \dots, R_M\}$ of the trees T_1 and T_2 . Then the distance formula is given as:

$$C_{T_1, T_2} = \sum_i^N \sum_j^M x_{ij} \omega_{ij} (d_{ij} I / L_{ij} + (1 - I) \delta_{ij} \epsilon_{ij} / 2) \quad (4.1)$$

This measure is calculated pairwise for all nodes K_i and R_j in two PSTs, namely each node in one tree is compared to the all other nodes in the other tree. Components of the formula are as follows.

$x_{ij} \in \{0, 1\}$ is 1 only if A_i is the prefix of B_j or vice versa and they are the closest such pair in length. This stands for assuring that either the same parameters/subsequences are compared to each other or in the case that a parameter/subsequence only exists in one model, then it is compared to the closest subsequence in the other tree. All the other comparisons in fact has zero contribution to total difference between two trees and in practice they are not done

(realization algorithm will be explained shortly), but they theoretically exist for consistency in the formula 1.

ω_{ij} keeps the average of the frequency of A_i and B_j being observed in A and B respectively: $\omega_{ij} = (P(A_i) + P(B_j))/2$. This weights the distance contributions of node comparisons in a way that the more subsequences of the compared nodes are observed in their corresponding sequences the more the contribution will be. Therefore, it prioritizes the information obtained from the sequences with more occurrences, namely with more witnesses.

I scales the contribution of two types of distance types: Probability distance and structural distance. The contribution of structural distance increases as I increases where contribution of probability distance decreases and vice versa.

d_{ij} holds the differences between the lengths of the subsequences A_i and B_j :

$$d_{ij} = \text{abs}(|A_i| - |B_j|) \quad (4.2)$$

where $|X|$ indicates the length of the subsequence X . The term is normalized by L_{ij} which is the larger of the two subsequences' lengths.

δ_{ij} is the main component of the probability distance calculation: it holds the absolute values of the element-wise differences in the probability vector of two nodes K_i and M_j :

$$\delta_{ij} = \sum_{k \in \Sigma} |(\overrightarrow{P_{A_i}})_k - (\overrightarrow{P_{B_j}})_k| \quad (4.3)$$

The term is normalized by 2 which is its maximum possible value of the difference between two probability vectors which happens when vectors have right angle between them.

$\epsilon_{ij} \in \{0, 1\}$ is 1 only if $A_i = B_j$. When 0, since the subsequences are not the same, a comparison of probabilities does not apply.

In **Enhanced PST Matching**, instead of ω_{ij} , W_{T_1, T_2} is introduced which is the total number of pairs of nodes that are compared in comparison of two PSTs that have a non-zero contribution (when x_{ij} is 1) to the total distance. Distance formula in Enhanced PSTM is given

as:

$$C_{T_1, T_2} = \frac{1}{W_{T_1, T_2}} \left(\sum_i^N \sum_j^M x_{ij} (d_{ij} I / L_{ij} + (1 - I) \delta_{ij} \epsilon_{ij} / 2) \right) \quad (4.4)$$

W_{T_1, T_2} normalizes each comparison of two PSTs by the total number of node comparisons done. This comes in handy in the following situation: consider two comparisons of two node pairs, namely in one hand trees T1 and T2 are compared which are the models of sequences A and B and in the other hand T3 and T4 are compared which are the models of the sequences C and D. Assume that the system which is the data obtained from has two type of behavior. Further assume that system has behavior 1 when sequences A and B are recorded and behavior 2 when C and D are recorded. If the total length of the sequences A and B is significantly larger than total length of C and D, then trees T1 and T2 would tend to have significantly higher number of nodes than that of the trees T3 and T4. In this case, even the distances calculated in these two comparisons are expected to be similar since in both comparisons, system manifests the same behavior in the data of compared periods, the distance values would be highly affected from the mentioned size difference. By introducing W_{T_1, T_2} , MTS Clustering is intended to work successfully even if the lengths of the MTSs in the data set are different.

We follow a procedure to realize EPSTM which operates in linear time with respect to the number of nodes in the larger PST as following. Starting from the root nodes of the compared PSTs a recursive procedure is applied so that, first, subsequences in the nodes are compared and if they are same, probability distance is calculated, if not, structural distance is calculated. Then the children of the nodes are traversed simultaneously. For all symbols in the alphabet, if a child that has a subsequence that starts with a specific symbol exists in both trees, these two children nodes are compared, if not, the node without the child and the child of the node with the child is compared. Naturally, for any node couple, if both compared nodes are the children of the nodes compared in the last step, probability distance will be calculated because their subsequences will be the same. If not, structural distance will be calculated since their subsequences will not be the same. Procedure is visualized in Figure 4.

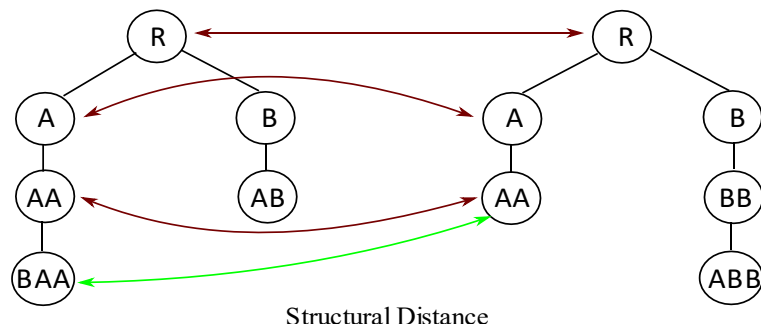
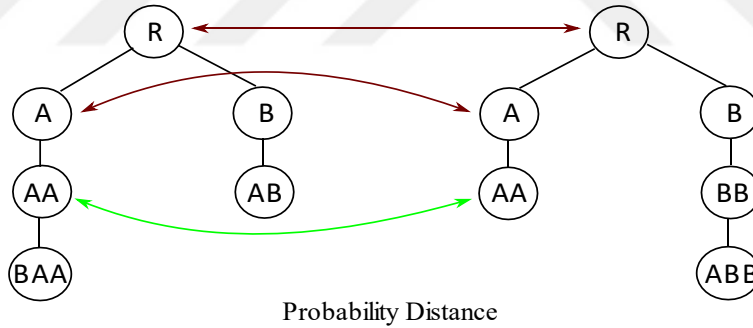
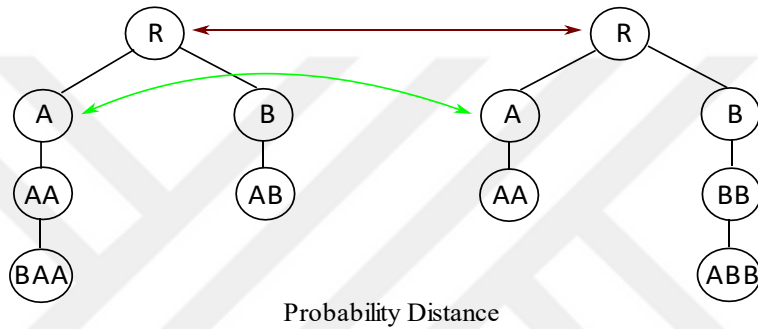
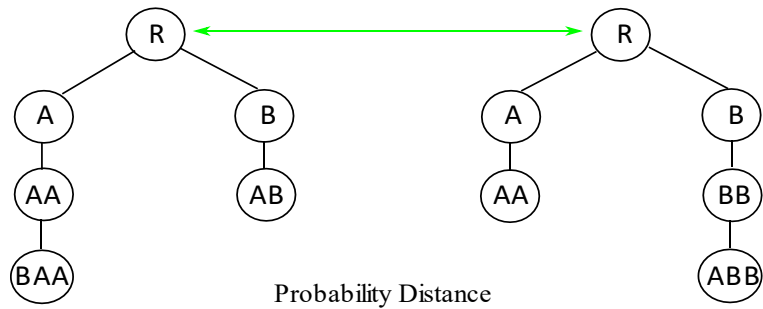


Figure 5: Visualization of the steps of PST Comparison (depth-first by recursion). If nodes with a specific sub-sequence exist in both trees, Probability Distance is calculated. If not, Structural Distance is calculated between the most relevant nodes. For each step, the nodes that are compared are shown with green arrows and the type of comparison is given below.

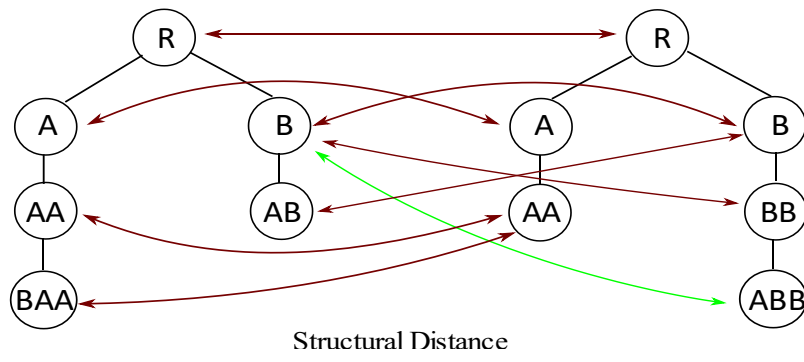
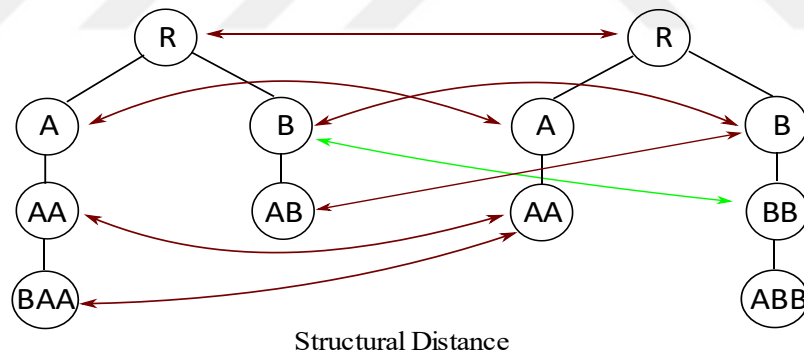
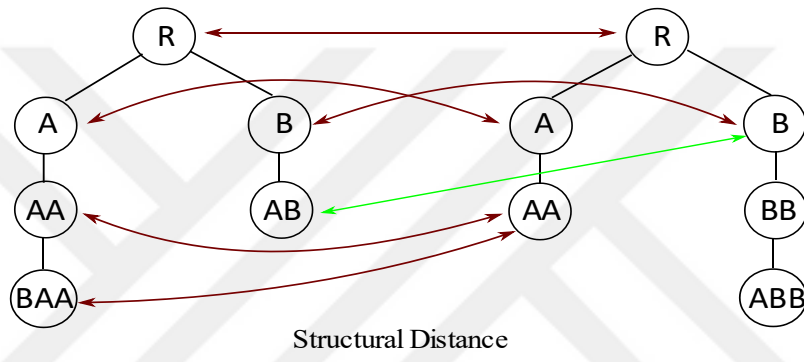
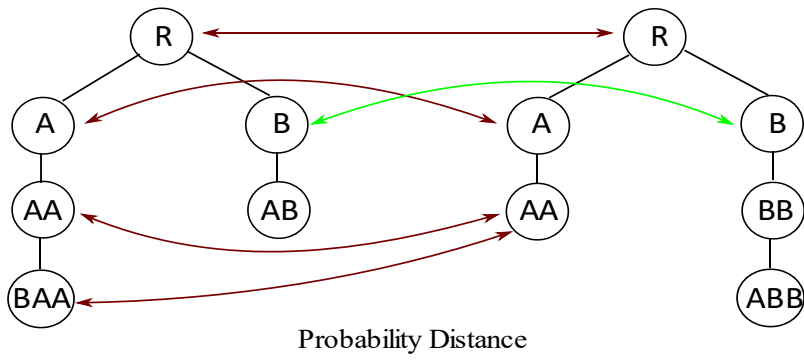


Figure 4 Continued

Pseudocode of the EPSTM algorithm is given below:

EPSTM(Root1, Root2, Alph)

1. If Root1.Subsequence = Root2.Subsequence
 1. Total Distance = Total Distance + I
2. Else:
 1. Total Distance = Total Distance + d
3. For i from 0 to len(Alph):
 1. If Root1.Children[i] != Null and Root2.Children[i] != Null
 - i. EPSTM(Root1.Children[i] , Root2.Children[i] , Alph)
 2. Else if Root1.Children[i] = Null and Root2.Children[i] != Null
 - i. EPSTM(Root1, Root2.Children[i] , Alph)
 3. Else if Root1.Children[i] != Null and Root2.Children[i] = Null
 - i. EPSTM(Root1.Children[i] , Root2, Alph)
 4. Else
 - i. Continue
4. Return

Frobenius Intersection: Frobenius Intersection, presented in [60] and shown to have superior success to PSTM on genomics data, is a more simplistic formulation that calculates only probability information differences between two PSTs, in a root mean square manner. A comparison is calculated only if the subsequences of the compared nodes are the same in both trees. It is a modification of the method presented in [61] which makes use of Frobenius Norm as a base method to compare two HMMs. Formulation is as follows:

$$FI_{T_1, T_2} = \sqrt{\frac{1}{|S_i|} \sum_{s \in S_i} \sum_{k \in \Sigma} \left((\overline{P_{A_i}})_k - (\overline{P_{B_j}})_k \right)^2} \quad (4.5)$$

VOMM MTS Clustering is visualized in figure 5.

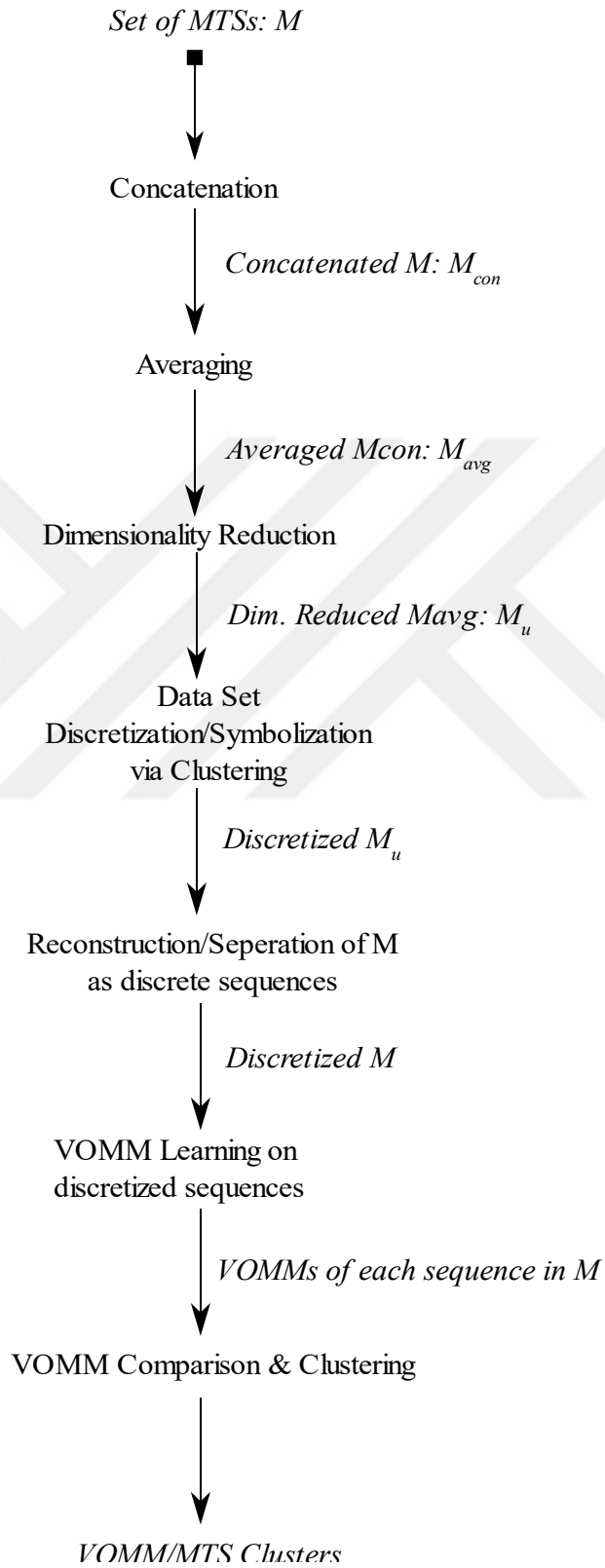


Figure 6: Steps of VOMM MTS Clustering

4.3 HMM MTS Modelling and Comparison

Several HMM based methods are proposed in MTS Analysis in the literature such as [18, 39, 62]. If the observable space is infinitely large as it is in MTS with d possibly continuous variables, it is impractical to learn an emission matrix which contains probabilities for each possible observable. To tackle this, a common approach which is also used in this study is to learn B as a set of multivariate Gaussian distributions that are fit on observable space, where each element corresponds to one hidden state of the HMM [63, 64].

For each MTS, an HMM is learned and the pairwise distances between these HMMs are calculated to obtain a dissimilarity matrix.

A collection of metrics that have been proposed in the literature to calculate a distance between pairwise HMMs are presented in [60]. Here we use a metric based on Frobenius matrix norm, which is used as a baseline metric in [61]. The idea and complexity are similar to those of the distance metrics presented in 4.2.3.

Frobenius norm is the root mean square of the elements of a matrix:

$$\|M\|_F = \sqrt{\sum_i \sum_j a_{ij}} \quad (4.6)$$

Then the distance between two HMMs λ_1 and λ_2 are the sum of Frobenius norms of the differences between their transition and emission matrices:

$$dF(\lambda_1, \lambda_2) = \|A_1 - A_2\| + \|B_1 - B_2\| \quad (4.7)$$

4.4 Principal Component Analysis MTS Modelling and Comparison

[17] proposed a PCA based MTSC method, which compares two MTS; MTS_1 and MTS_2 , by two criteria, (1) the pairwise angle θ_{ij} between the first k principal components and (2) the spatial distance between the data points.

The contribution of the angle between the principal components are calculated as follows:

$$S_{pca} = \frac{\sum_{i=1}^k \sum_{j=1}^k (\lambda_i^{(1)} \lambda_j^{(2)}) \cos \theta_{ij}}{\sum_{i=1}^k \sum_{j=1}^k \lambda_i^{(1)} \lambda_j^{(2)}} \quad (4.8)$$

To weight the contribution of the distance between the principal components, the PCs are weighted with their explained variance. This is done by multiplying the eigenvalues of the PCs being compared since the explained variance of a PC is proportional to the corresponding eigenvalue. $\lambda^{(1)}$ and $\lambda^{(2)}$ are the i 'th eigenvalues of the covariance matrices of MTS_1 and MTS_2 respectively.

S_{pca} only takes into account the spatial orientation of the data but it is inadequate in the situations where the spatial orientation of the data matrices is similar, but the data points are located far in the data space. Therefore, a spatial distance formulation which is originally defined in [17] is given as follows:

$$S_{dist} = 2 \times \left[1 - \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\Phi} e^{-z^2/2} dz \right] \quad (4.9)$$

Where:

$$\Phi = (\bar{x}_2 - \bar{x}_1) \Sigma_1^{*-1} (\bar{x}_2 - \bar{x}_1)^T \quad (4.10)$$

\bar{x}_1 and \bar{x}_2 are sample mean row vectors and Σ_1 is the covariance matrix for dataset MTS_1 and Σ_1^{-1} is the pseudo-inverse of MTS_1 calculated using singular value decomposition.

4.5 MTS Clustering

Based on the obtained dissimilarity matrix, clustering is applied to the models to classify the MTS. Experiments are done using three different clustering methods, (1) k-means, (2) Hierarchical DBSCAN [65], (3) k-medoids. Our experiments showed that k-means method does not perform well and gives unstable results even with high number of iterations. A complex method such as HDBSCAN took too many iterations to converge to a stable clustering result and also had poor robustness to the parameters. k-medoids method performed the best out of these three clustering methods and experiments showed that it is

also easier to automatically determine the optimal/suboptimal clustering parameters (see section 5). To limit the scope of this study, a detailed description or experimental evaluation for k-means and Hierarchical DBSCAN methods will not be given.

K-medoids chooses data points as centers (medoids or exemplars) and minimizes an arbitrary metric of distance between these centers and the points assigned to clusters. A medoid can be defined as the object of a cluster whose average dissimilarity to all the objects in the cluster is minimal, i.e., it is the most centrally located point in the cluster. We use a common realization of k-medoids approach, Partitioning Around Medoids algorithm [66].

5 COMPLEXITY ANALYSIS

Preprocessing: If n is the average number of data points in the MTSs in the data set and r is the number of MTSs, averaging step takes $O(nr)$ time since it can be done by accumulating the data point values and dividing them by P in a single run on the data set. It is shown in [67] that PCA can be applied approximately in $O(d^2nr)$ to reduce the dimensionality from d to u .

VOMM MTS Learning and Comparison: Agglomerative Clustering method for data point clustering has the complexity of $O(u(nr)^2)$ [56]. For all this steps up to this point, $O(nr)$ can be substituted by $O(N)$ where N is the size of the complete data set since these steps are applied on all data points regardless of which MTS they belong to. VOMM Learning is dominated by the subsequence extraction phase (see 4.2.2) which has the complexity of $O(nL)$ where n is the size of the MTS and the corresponding discrete sequence that VOMM represents and L is the maximum order that is allowed in the VOMM. This is so since sorting operation is done L times where each sorting takes $O(n)$ time. Construction of models of all MTS therefore takes $O(NL)$. VOMM comparison can be done by one to one comparison of each node pair in both trees, therefore the complexity can be written as $O(g/\Sigma/)$ where g is the average number of nodes in one VOMM and $|\Sigma/$ is the alphabet size. VOMMs are claimed to be concise which is enabled by pruning and therefore, $g \ll n$ in practice. Therefore, the complexity of VOMM comparison is $O(n/\Sigma/)$. The calculation of the dissimilarity matrix takes $O(nr^2/\Sigma/)$. We know that $L \ll N$, then the complexity of VOMM MTS learning and comparison is $O(d^2N + uN^2 + r^2n/\Sigma/)$.

HMM MTS Learning and Comparison: Baum-Welch algorithm that has the complexity of $O(n/\Sigma/2)$ for each iteration [6] where $|\Sigma/$ is the number of hidden states. When MTS is the input and the emissions are learned as Multivariate Gaussian distributions, complexity is dominated by calculation of the covariance matrices if $|\Sigma/ < d$. This is so since calculating the covariance matrix requires $m^T m$ operation, which has the complexity of $O(u^2n)$. If we call the number of iterations needed for convergence W , then the complexity of learning the HMMs of all MTSs in the dataset is $O(u^2WN)$. Naturally, W is expected to be larger as the

size of the data set grows. Comparing a pair of HMMs with Frobenius Norm is $O(u^2/|\Sigma|)$ where $|\Sigma|$ is the number of hidden states, since there is one covariance matrix for each state which has the shape $u \times u$. Accordingly, calculating the dissimilarity matrix takes $O(r^2 d^2 / |\Sigma|)$. In total HMM MTS learning and comparison takes $O(d^2 N + u^2 W N + r^2 u^2 / |\Sigma|)$.

PCA MTS Learning and Comparison: If the complexity of applying PCA on a MTS is $O(d^2 n)$, then applying PCA on all MTSs will take $O(d^2 N)$. If the data is projected on a u dimensional space, calculation of S_{pca} between two MTS PCA models will take $O(u^3)$ since; the cosine of the angle of two vectors is the quotient of their dot product that takes $O(u)$ to calculate and cosine operation is done pairwise for all the covariance matrices of the MTSs. Computation of Φ in S_{dist} is dominated by the calculation of pseudo-inverse of Σ that is done by Singular Value Decomposition which has the complexity of $O(u^2 d)$ [68]. Since $u^2 d > u^3$, the computation of the dissimilarity matrix takes $O(d^2 N + r^2 u^2 d)$.

MTS Clustering: Finally, k-medoids takes $O(k(r-k)^2)$ for each iteration where one should note that $r \ll n$ in general case.

The terms that make the difference in three approaches are $uN^2 + r^2 n / |\Sigma|$, $u^2 W N + r^2 u^2 / |\Sigma|$ and $r^2 u^2 d$ respectively. the last term is highly unlikely to be larger than the first two in general unless the data dimensionality is extremely high. Therefore, we can say that PCA MTSC is the most efficient method in time. The second parts of the first two terms are expected to have no large difference and in fact the terms are dominated by their first parts with the existence of N . If Baum-Welch algorithm (see section 2.5) converges in few iterations as in section 6 then $uW < N$ will hold and we can conclude that HMM MTSC is practically more efficient than VOMM MTSC. Nevertheless, it is possible to use only a subset of the dataset for the preprocessing and data point clustering (see section 4.2.1) steps as explained in Appendix B which can drastically decrease the complexity of VOMM MTSC.

6 EXPERIMENTAL EVALUATION

6.1 Lego Demonstrator Data

The Lego Demonstrator is shown in Figure 6. A Lego piece is carried from its initial position in the magazine to the conveyor belt and placed on one of the two sticks.

One *run* of the demonstrator uses six input Lego pieces to produce two products on the two sticks on the conveyor belt, by stacking three of the incoming pieces to the first stick and the other three incoming pieces to the second stick on the conveyor belt. Input pieces are processed sequentially. To simulate product variation in the system, the order of sticks that are used for placing the input piece is varied. For example, if we name one of the sticks as 1 and the other one as 2, one run may have the stick sequence $1 \rightarrow 1 \rightarrow 2 \rightarrow 1 \rightarrow 2 \rightarrow 2$ to place all six pieces.

This setup produces a sensor and actuator data sequence similar to a real industrial plant. Concretely for each run we obtain a MTS log consisting of voltage and current values for two control units (Lego Bricks), sensor output one touch sensor used for stick alignment, moreover motor information (speed, angle, and motor command) for five motors. Overall this yields 20 signals that are logged each 250 ms: 2 real, 12 integer, and 6 binary signals. The data used in experiments in this paper was obtained from 107 runs. Each run moves 6 Lego pieces and produces 2 products.

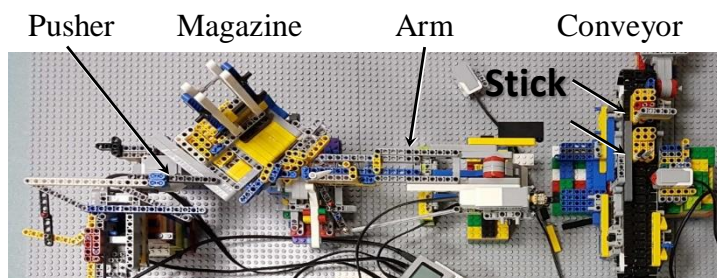


Figure 7: Lego Demonstrator Photo

Labels: Production sequences (i.e., the order of sticks) are logged and used as the true labels of the MTSs. We use three distinct product types where the underlying sequence is $2 \rightarrow 1 \rightarrow 2 \rightarrow 1 \rightarrow 2 \rightarrow 1$, $2 \rightarrow 2 \rightarrow 2 \rightarrow 1 \rightarrow 1 \rightarrow 1$ and $2 \rightarrow 1 \rightarrow 2 \rightarrow 2 \rightarrow 1 \rightarrow 1$ respectively, and we performed 36, 36, 35 runs for each product type.

Table 2: List of all parameters for all methods and their settings used for experiments.

Method	Prm	Description	Settings	
			Demonst.	Arcelik
Preprocessing	a	Averaging parameter, data frame size to be averaged.	2,3 ... 15	
	d	The number of dimensions the data reduced to by PCA (applied on the whole data set).	7, 10, 15	4, 5, 6
VOMM MTS Modelling and Comparison	k	Number of clusters for data point clustering, see 4.2.1.	2,3, ... 10	
	t	Minimum support parameter, see 4.2.2.	1,2, ... 9	
	L	Maximum subsequence length parameter, see also 4.2.2.	1,2, ... 9	
	I	Ratio of contributions of dissimilarity and probability distance, see 4.2.3 (Only for Enhanced PSTM).	0.1, 0.2, ... 1	
HMM MTS Modelling and Comparison	S	The number of hidden states to be modelled, see 2.6.	2, 3, ... 10	
	W	Number of iterations of the Baum-Welch algorithm. It is more probable to achieve higher levels of convergence with higher number of iterations.	2, 4, 6, 8, 10, 25, 50, 100, 1000, 1200, 1400, 1600, 1800, 2000	
PCA MTS Modelling and Comparison	d	The number of dimensions the data reduced to by PCA (applied to each MTS seperately) see 4.4.	7, 10, 15	4, 5, 6
	α	Ratio of contributions of two distance types S_{pca} and S_{dist} , see 4.4.	0.1, 0.2, ... 1	
Model Clustering	mk	Number of clusters for model clustering, see 4.5.	2, 3, 4, 5, 6	

6.2 Arçelik Hydraulic Press Machine Data

Arçelik Hydraulic Press Machine is one of the key components of the dishwasher plants of Arçelik. It shapes the metal plates with pressure which are then used to cover the cage of

the dishwasher machine. Machine can stop due to many reasons; planned stoppages such as holidays or because there is an anomaly. Sensor data is continuously recorded with six different variables such as pressure, oil level and temperature. Many types of anomalies may occur as the machine is running such as electrical or mechanical anomalies etc. Each of the stoppages are recorded as either planned stoppage or anomaly with the root cause. These periods are used as MTSs, and the stoppage reason information are used as the true labels for applying MTS Clustering. The data set we used in the experiments is from one week of life time of the dishwasher plant. The sensor data of the periods that the machine is running are the MTSs to be clustered and the reasons of the stoppages after that period are used as the labels. This setup is visualized in Figure 7.

In total there are 56 MTSs where 51 of them are recorded when the system behavior is expected to be normal, and 5 are recorded in anomalous periods of 3 type; 1 mechanical errors, 1 electrical and 3 mold error.

6.3 Setup

The parameters and their settings for each method are given in Table 2. For each method and data set, we perform experimental runs for all combinations of all the shown parameter values.

Predicted clusters are compared with real clusters of VOMMs and scores are calculated as

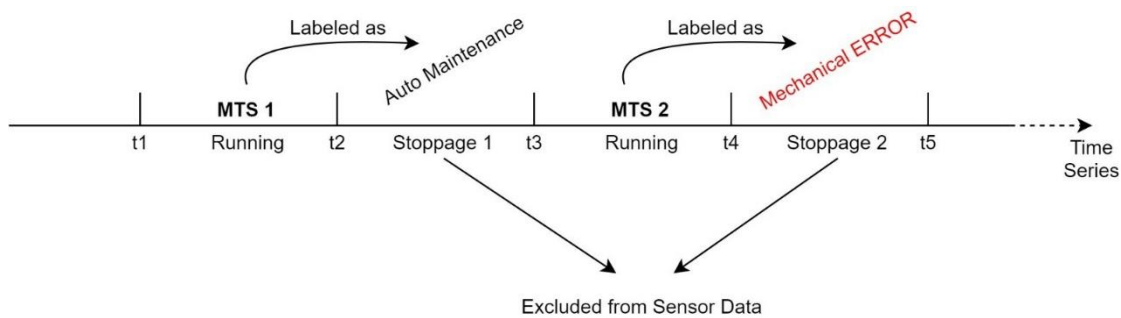


Figure 8: Demonstration of the setup of the Arcelik Hydraulic Press Machine Data

Adjusted Rand Index [69] which is a corrected-for-chance version of the commonly used clustering metric Rand Index. Rand Index score of clustering result is scaled with the average success of a random clustering method (expected score).

Given a set S of n elements and two clusterings: $C = \{C_1, C_2, \dots, C_r\}$, $D =$

$\{D_1, D_2, \dots, D_s\}$ of these elements, the overlap between C and D can be summarized in a contingency table $[n_{ij}]$ where each entry denotes the number of objects in common between C_i and D_j : $n_{ij} = |X_i \cap Y_j|$. Given that $a_i = n_{i1} + n_{i2} + \dots + n_{is}$ and $b_j = n_{1j} + n_{2j} + \dots + n_{sj}$, adjusted rand index is defined as:

$$ARI = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2} \right] / \binom{n}{2}}{\frac{1}{2} \left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2} \right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2} \right] / \binom{n}{2}} \quad (5.1)$$

6.4 Results

As it can be seen from the Tables 7 and 8, VOMM MTSC method with Enhanced PST Matching distance achieves up to 0.75 ARI on Marmara demonstrator data and achieves up to 0.61 ARI on Arçelik hydrolic machine data. VOMM MTSC with Frobenius Intersection distance results with 0.64 to 0.41 ARIs respectively. HMM MTSC cannot achieve a significantly higher success than a random clustering for Lego Demonstrator data with at highest 0.06 ARI, but it can achieve up to 0.29 ARI on Arçelik data, see Table 5 & 6. PCA MTSC did not perform better than a random clustering, at highest 0.04 ARI on demonstrator data and 0.03 on Arçelik data were achieved, see Table 3 & 4.

Combinations of parameter values are shown in the result tables, other combinations of the values either give worse results or there is no significant difference from the closest shown value in the table. About results regarding to parameter settings we can say the following.

VOMM MTSC: Since Demonstrator data has three different types of MTS, naturally best results were obtained with $mk = 3$. For Arçelik data there are four different types of MTS and best results are obtained with mk values 3, 4 and 5 correspondingly. Best results are achieved with $k = 2$ and $k = 3$ on the Lego Demonstrator data set and on Arçelik Press Machine data set, k values 5, 6 and 7 gave the best results. We see that d does not have a high impact on the results with the settings mentioned above, but experiments on Demonstrator Data showed that using whole the variance achieved slightly worse results. The best values for a was 6 and 7 for Demonstrator Data where for Arçelik Data even 8, 9 and 10 achieved highest results, no

significant difference is observed with other values. For t , on both data sets we see that range of values 4, 5, 6 gives the best results and there is no significant success difference between these settings. The success slightly increases as the L value increases up to 7 for both data sets and higher values give similar results to this value. VOMM MTSC with Frobenius Intersection distance showed very similar responses to the parameter settings mentioned up this point and therefore a separate table of results for it will not be given. Regarding I for EPSTM distance, for Demonstrator data only $I = 0.5$ is shown and values between 0.3 – 0.7 gave exactly the same results. For Arçelik data, $I = 0.1$ gave best results and the other values gave worse results.

Table 3: Adjusted Rand Index Results of PCA MTS Clustering done with the combinations of parameters explained in table 2 on Demonstrator Data. Adjusted RAND Index (ARI) is used for scoring.

a	d	$\alpha =$	0.3				0.4				0.5				0.6				0.7			
		mk=	2	3	4	5	2	3	4	5	2	3	4	5	2	3	4	5	2	3	4	5
1	3		-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00
	5		-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00
	7		-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00
	10		-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00
	15		-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00
2	3		-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00
	5		-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01
	7		-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00
	10		-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00
	15		-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00
3	3		-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01
	5		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
	7		0.01	0.01	0.00	0.00	0.01	0.01	0.00	0.00	0.01	0.01	0.00	0.00	0.01	0.01	0.00	0.00	0.01	0.01	0.00	0.00
	10		0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03	0.03
	15		0.04	0.03	0.03	0.03	0.04	0.03	0.03	0.03	0.04	0.03	0.03	0.03	0.04	0.03	0.03	0.03	0.04	0.03	0.03	0.03
4	3		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
	5		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
	7		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
	10		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
	15		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
5	3		0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	
	5		0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02
	7		0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02
	10		0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02
	15		0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02	0.01	0.01	0.01	0.02
6	3		0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
	5		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
	7		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
	10		0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00
	15		0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00
7	3		0.00	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.00	0.03	0.03	0.03	0.00	0.03	0.03	0.03
	5		0.00	0.01	0.01	0.01	0.00	0.01	0.01	0.01	0.00	0.01	0.01	0.01	0.00	0.01	0.01	0.01	0.00	0.01	0.01	0.01
	7		-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01
	10		0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.00
	15		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	3		0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
	5		-0.01	0.00	0.00	0.01	-0.01	0.00	0.00	0.01	-0.01	0.00	0.00	0.01	-0.01	0.00	0.00	0.01	-0.01	0.00	0.00	0.01
	7		-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01
	10		-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00
	15		-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00
9	3		0.00	0.01	0.01	0.00	0.00	0.01	0.01	0.00	0.00	0.01	0.01	0.00	0.00	0.01	0.01	0.00	0.00	0.01	0.01	0.00
	5		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
	7		0.00	-0.01	-0.01	0.00	0.00	-0.01	-0.01	0.00	0.00	-0.01	-0.01	0.00	0.00	-0.01	-0.01	0.00	0.00	-0.01	-0.01	0.00
	10		0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.01
	15		0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00
10	3		0.00	0.00	-0.01	-0.01	0.00	0.00	-0.01	-0.01	0.00	0.00	-0.01	-0.01	0.00	0.00	-0.01	-0.01	0.00	0.00	-0.01	-0.01
	5		-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00
	7		-0.01	0.00	0.01	0.01	-0.01	0.00	0.01	0.01	-0.01	0.00	0.01	0.01	-0.01	0.00	0.01	0.01	-0.01	0.00	0.01	0.01
	10		-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.00
	15		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

HMM MTSC: By analyzing Table 6, we observe that the success increases by using higher number of hidden states up to $S = 4$, but there is no improvement with $S > 4$. Best results are

achieved at $mk = 2$ since actually HMM MTSC could distinguish only one of the anomalous MTS and the rest were clustered in the same cluster with normal MTS. Here PCA did not have a high impact on the results. Averaging also had little impact on the success and $ds = 9$ and $a = 10$ gave the best results, while other settings gave slightly worse results. We also see that at $W = 8$ the model parameters already converge regardless of other hyper-parameter settings.

Table 4: Adjusted Rand Index Results of PCA MTS Clustering done with the combinations of parameters explained in table 2 on Arcelik Data. Adjusted RAND Index (ARI) is used for scoring.

a	d	$\alpha=$	0.3				0.4				0.5				0.6				0.7							
		mk=	2	3	4	5	2	3	4	5	2	3	4	5	2	3	4	5	2	3	4	5				
1	3		0.02	0.02	0.01	0.01	0.02	0.02	0.01	0.01	0.02	0.02	0.01	0.01	0.02	0.02	0.01	0.01	0.02	0.02	0.01	0.01	0.02	0.02	0.01	0.01
	4		0.01	0.01	0.01	0.00	0.01	0.01	0.01	0.00	0.01	0.01	0.01	0.00	0.01	0.01	0.01	0.00	0.01	0.01	0.01	0.00	0.01	0.01	0.01	0.00
	5		0.02	0.01	0.01	0.00	0.02	0.01	0.01	0.00	0.02	0.01	0.01	0.00	0.02	0.01	0.01	0.00	0.02	0.01	0.01	0.00	0.02	0.01	0.01	0.00
	6		0.02	0.01	0.01	0.00	0.02	0.01	0.01	0.00	0.02	0.01	0.01	0.00	0.02	0.01	0.01	0.00	0.02	0.01	0.01	0.00	0.02	0.01	0.01	0.00
2	3		-0.03	0.01	0.01	0.01	-0.03	0.01	0.01	0.01	-0.03	0.01	0.01	0.01	-0.03	0.01	0.01	0.01	-0.03	0.01	0.01	0.01	-0.03	0.01	0.01	0.01
	4		-0.01	-0.01	-0.02	-0.02	-0.01	-0.01	-0.02	-0.02	-0.01	-0.01	-0.02	-0.02	-0.01	-0.01	-0.02	-0.02	-0.01	-0.01	-0.02	-0.02	-0.01	-0.01	-0.02	-0.02
	5		-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.02	-0.01	-0.01	-0.01	-0.02
	6		0.00	-0.01	-0.01	-0.02	0.00	-0.01	-0.01	-0.02	0.00	-0.01	-0.01	-0.02	0.00	-0.01	-0.01	-0.02	0.00	-0.01	-0.01	-0.02	0.00	-0.01	-0.01	-0.02
3	3		-0.03	-0.04	-0.04	-0.05	-0.03	-0.04	-0.04	-0.05	-0.03	-0.04	-0.04	-0.05	-0.03	-0.04	-0.04	-0.05	-0.03	-0.04	-0.04	-0.05	-0.03	-0.04	-0.04	-0.05
	4		-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04
	5		-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04
	6		0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01
4	3		-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04
	4		-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04
	5		-0.03	-0.04	-0.04	-0.05	-0.03	-0.04	-0.04	-0.05	-0.03	-0.04	-0.04	-0.05	-0.03	-0.04	-0.04	-0.05	-0.03	-0.04	-0.04	-0.05	-0.03	-0.04	-0.04	-0.05
	6		0.01	-0.01	-0.01	-0.02	0.01	-0.01	-0.01	-0.02	0.01	-0.01	-0.01	-0.02	0.01	-0.01	-0.01	-0.02	0.01	-0.01	-0.01	-0.02	0.01	-0.01	-0.01	-0.02
5	3		-0.03	-0.01	-0.01	-0.02	-0.03	-0.01	-0.01	-0.02	-0.03	-0.01	-0.01	-0.02	-0.03	-0.01	-0.01	-0.02	-0.03	-0.01	-0.01	-0.02	-0.03	-0.01	-0.01	-0.02
	4		-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04	-0.03	-0.04	-0.04	-0.04
	5		0.01	0.00	-0.01	-0.01	0.01	0.00	-0.01	-0.01	0.01	0.00	-0.01	-0.01	0.01	0.00	-0.01	-0.01	0.01	0.00	-0.01	-0.01	0.01	0.00	-0.01	-0.01
	6		0.01	0.00	0.00	-0.01	0.01	0.00	0.00	-0.01	0.01	0.00	0.00	-0.01	0.01	0.00	0.00	-0.01	0.01	0.00	0.00	-0.01	0.01	0.00	0.00	-0.01
6	3		-0.03	-0.01	-0.02	-0.02	-0.03	-0.01	-0.02	-0.02	-0.03	-0.01	-0.02	-0.02	-0.03	-0.01	-0.02	-0.02	-0.03	-0.01	-0.02	-0.02	-0.03	-0.01	-0.02	-0.02
	4		-0.03	-0.02	-0.02	-0.02	-0.03	-0.02	-0.02	-0.02	-0.03	-0.02	-0.02	-0.02	-0.03	-0.02	-0.02	-0.02	-0.03	-0.02	-0.02	-0.02	-0.03	-0.02	-0.02	-0.02
	5		0.01	-0.02	-0.02	-0.02	0.01	-0.02	-0.02	-0.02	0.01	-0.02	-0.02	-0.02	0.01	-0.02	-0.02	-0.02	0.01	-0.02	-0.02	-0.02	0.01	-0.02	-0.02	-0.02
	6		0.01	-0.01	-0.02	-0.02	0.01	-0.01	-0.02	-0.02	0.01	-0.01	-0.02	-0.02	0.01	-0.01	-0.02	-0.02	0.01	-0.01	-0.02	-0.02	0.01	-0.01	-0.02	-0.02

Table 5: Adjusted Rand Index Results of HMM MTS Clustering done with the combinations of parameters explained in table 2 on Demonstrator Data. Adjusted RAND Index (ARI) is used for scoring.

S	W	mk	a =	1			2			3			4			5			
			d =	7	10	15	7	10	15	7	10	15	7	10	15	7	10	15	
2	1000	2		-0.01	-0.01	-0.01	0.00	0.01	0.00	0.05	0.00	0.03	0.00	0.00	0.00	-0.01	-0.01	0.01	
		3		-0.01	-0.01	0.01	0.00	0.01	0.00	0.03	-0.01	0.01	0.00	-0.01	0.00	-0.01	-0.01	0.01	
		4		-0.01	0.00	0.01	-0.01	0.01	-0.01	0.02	-0.01	0.01	0.00	0.00	0.01	-0.01	0.00	0.01	
		5		-0.01	0.00	0.01	0.00	0.01	-0.01	0.01	-0.01	0.00	0.00	0.00	0.00	0.01	-0.01	0.00	0.00
	1200	2		-0.01	-0.01	0.00	-0.01	0.01	0.01	0.01	0.01	0.02	-0.01	0.01	-0.01	-0.01	0.04	0.01	
		3		-0.01	-0.01	-0.01	-0.01	0.01	0.00	-0.01	0.00	0.00	-0.02	0.00	-0.01	-0.01	0.05	0.02	
		4		-0.01	-0.01	-0.01	-0.01	0.00	0.00	-0.01	0.00	0.00	-0.01	0.00	-0.01	-0.01	0.05	0.00	
		5		-0.01	-0.01	-0.01	-0.01	0.00	0.00	-0.01	-0.01	0.00	-0.01	0.00	-0.01	-0.01	0.05	0.01	
	1400	2		0.00	-0.01	-0.01	-0.01	-0.01	-0.01	0.02	0.02	0.02	0.00	0.00	0.00	-0.01	-0.01	0.00	
		3		0.00	-0.01	-0.01	-0.01	-0.01	-0.01	0.03	0.01	0.01	-0.01	0.00	0.00	-0.01	0.00	0.01	
		4		-0.01	-0.01	-0.01	-0.01	0.00	0.00	0.03	0.01	0.01	-0.01	0.00	0.00	-0.01	0.00	0.01	
		5		-0.01	-0.01	-0.01	-0.01	0.00	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
	1600	2		-0.01	-0.01	0.00	-0.01	0.02	-0.01	0.01	0.02	0.04	0.00	-0.01	-0.01	0.00	-0.01	-0.01	
		3		-0.01	-0.01	-0.01	-0.01	0.03	0.00	0.01	0.06	0.03	0.00	0.00	-0.01	0.00	0.00	0.00	
		4		-0.01	-0.02	-0.01	-0.01	0.03	0.00	0.01	0.06	0.02	0.00	0.00	0.00	-0.01	0.01	0.00	
		5		-0.01	-0.02	-0.01	-0.01	0.03	-0.01	0.01	0.06	0.02	0.00	0.00	0.00	0.00	0.00	0.00	
	3	1000	2		-0.01	-0.01	-0.01	0.00	-0.01	-0.01	0.00	0.01	0.03	0.02	0.00	0.00	0.01	0.02	0.00
			3		-0.02	0.00	0.00	0.00	-0.01	-0.01	0.00	0.01	0.03	0.01	0.00	0.00	0.01	0.01	0.00
			4		-0.01	-0.01	0.01	0.00	0.00	-0.01	0.02	0.01	0.02	0.00	-0.01	-0.01	0.00	0.00	-0.01
			5		-0.01	-0.01	0.01	0.00	0.00	-0.01	0.02	0.00	0.01	0.01	-0.01	-0.01	0.00	0.00	0.00
1200		2		-0.01	-0.01	0.01	-0.01	0.00	0.03	0.02	0.01	0.01	0.02	-0.01	0.01	0.02	0.02	0.01	
		3		-0.02	-0.01	0.01	-0.01	-0.01	0.02	0.02	0.01	0.00	0.01	-0.01	0.00	0.01	0.02	0.01	
		4		-0.02	-0.01	0.01	-0.01	-0.01	0.03	0.01	0.02	0.02	0.01	-0.01	0.00	0.01	0.01	0.00	
		5		-0.02	0.00	0.01	-0.01	-0.01	0.03	0.00	0.00	0.01	0.00	-0.01	0.01	0.02	0.01	0.02	
1400		2		-0.01	0.00	0.00	0.00	-0.01	-0.01	-0.01	0.01	0.02	0.00	-0.01	0.00	0.01	0.01	0.01	
		3		-0.01	0.01	0.00	-0.01	-0.01	-0.01	0.00	0.02	0.01	0.00	0.00	-0.01	0.01	0.00	0.01	
		4		-0.01	0.00	0.00	0.00	-0.01	0.00	0.00	0.02	0.01	-0.01	-0.01	0.00	0.00	-0.01	0.00	
		5		0.00	0.00	-0.01	0.00	-0.01	-0.01	0.01	0.02	0.00	-0.01	-0.01	0.00	0.00	-0.01	0.00	
1600		2		0.00	-0.01	0.01	0.00	-0.01	-0.01	0.04	-0.01	-0.01	0.00	0.00	0.00	0.00	0.02	0.02	
		3		0.00	-0.01	0.02	0.01	-0.01	-0.01	0.04	0.00	0.00	0.00	0.00	0.01	0.00	0.02	0.02	
		4		-0.01	-0.01	0.00	0.00	-0.01	-0.01	0.03	0.00	0.00	0.00	-0.01	0.01	-0.01	0.02	0.02	
		5		-0.01	-0.01	0.00	0.00	-0.01	0.01	0.01	0.00	0.00	0.00	0.00	0.00	-0.01	0.04	0.00	
4		1000	2		-0.01	-0.01	-0.01	-0.01	-0.01	0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.00	-0.01	0.02	0.00
			3		0.00	0.00	-0.01	-0.01	-0.01	0.00	-0.01	0.01	0.00	-0.01	-0.01	-0.01	0.00	0.01	-0.01
			4		0.00	0.00	-0.01	-0.01	0.00	0.00	-0.01	0.02	0.02	-0.01	0.01	-0.01	-0.01	0.01	-0.01
			5		0.00	0.00	-0.01	0.00	0.00	-0.01	0.00	0.01	0.02	-0.02	0.00	0.00	-0.01	-0.01	0.00
	1200	2		0.00	0.00	0.01	-0.01	0.00	0.00	0.01	0.00	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	
		3		0.00	-0.02	0.00	0.00	-0.01	-0.01	0.02	0.00	0.00	0.01	0.00	0.00	0.01	-0.01	-0.01	
		4		0.01	-0.02	0.00	0.00	-0.02	-0.01	0.02	0.00	0.02	0.01	0.00	0.01	0.01	0.00	0.01	
		5		0.01	-0.01	-0.01	-0.01	-0.02	-0.01	0.02	-0.01	0.02	0.00	-0.01	0.01	0.03	-0.01	0.01	
	1400	2		0.00	-0.01	-0.01	0.00	-0.01	-0.01	0.00	0.01	0.00	0.00	0.00	0.00	-0.01	-0.01	0.00	
		3		0.00	0.00	-0.01	0.00	-0.01	-0.01	0.00	0.02	0.00	0.00	0.00	0.00	-0.01	0.00	0.00	
		4		0.00	0.00	-0.01	0.01	-0.01	-0.01	0.00	0.01	0.00	0.00	-0.01	0.00	-0.01	0.00	0.00	
		5		0.00	0.02	-0.01	0.01	-0.01	0.00	0.00	0.00	0.00	-0.01	-0.01	0.00	0.00	0.01	-0.01	
	1600	2		-0.01	0.00	0.02	0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	0.03	-0.01	-0.01	0.00	-0.01	
		3		0.02	0.00	0.01	0.00	-0.01	-0.01	0.00	-0.01	0.00	-0.01	0.03	-0.01	0.02	-0.01	0.00	
		4		0.01	0.01	0.01	0.00	-0.01	-0.01	0.00	0.00	0.00	-0.01	0.03	-0.01	0.01	0.00	0.01	
		5		0.01	0.00	-0.01	0.00	0.00	0.00	0.00	0.00	0.00	-0.01	0.03	0.00	0.00	0.00	0.02	

6.5 Discussion

Success of VOMM with $k = 2$ with Demonstrator data can be explained by the clusters that consist of the data points that corresponds to placements on two different sticks, see 6.1. Arçelik Hydraulic Press Machine arm has 3 degrees of freedom which can be related to the success on Arçelik data with k values around 6. PCA yields noise elimination in variance up to some degree and therefore d values 7, 10 and 15 gives better results than $d = 26$. No such contribution observed in Arçelik data which may be due enterprise quality sensors tend to record the data with lower noise rate. We observe that t values lower than 4 is not high enough for pruning away the noise while using higher values than 6 can cause loss of information. Another important observation is that appropriate settings of t make the method more robust to the averaging. For both data sets, when t is higher than 6, even higher information loss occurs as a increases to the values higher than 8. On the other hand, when t is lower than 4, high values of a makes the significant information mix up with the noise. Naturally, as L increases longer sub-sequences can be kept in the model which allows characterization of higher detail (very high values bring potential of overfitting). I seems to have very little effect on the results on Demonstrator data. While in Arçelik data, higher values than 0.1 decreases the results to 0.29 which is the highest score achieved with HMM (same clustering structure is obtained).

Table 6: Adjusted Rand Index Results of HMM MTS Clustering done with the combinations of parameters explained in table 2 on Demonstrator Data. Adjusted RAND Index (ARI) is used for scoring.

S	W	mk	a=	6			7			8			9			10		
			d=	4	5	6	4	5	6	4	5	6	4	5	6	4	5	6
2	4	2		0.02	0.03	0.03	0.02	0.02	0.02	-0.01	0.00	-0.01	0.00	0.00	0.00	-0.01	-0.01	0.00
		3		-0.01	-0.03	-0.03	-0.02	-0.02	-0.02	0.00	0.00	0.00	0.01	0.01	0.00	0.01	0.02	0.04
		4		-0.02	-0.05	-0.04	-0.01	-0.02	0.00	-0.01	0.00	-0.02	0.01	0.01	0.00	0.00	0.00	0.01
	6	2		0.02	0.03	0.03	0.02	0.02	0.02	-0.01	0.00	-0.01	0.00	0.00	0.00	-0.01	-0.01	0.00
		3		-0.01	-0.03	-0.03	-0.02	-0.02	-0.02	0.00	0.00	0.00	0.01	0.01	0.00	0.02	0.02	0.04
		4		-0.02	-0.04	-0.04	-0.01	-0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.02
	8	2		0.02	0.03	0.04	0.02	0.02	0.03	-0.01	0.00	-0.01	0.00	0.00	0.00	-0.01	-0.01	0.00
		3		-0.01	-0.03	-0.03	-0.02	-0.02	-0.02	0.00	0.00	0.00	0.01	0.01	0.00	0.02	0.02	0.04
		4		-0.02	-0.04	-0.04	-0.01	-0.02	-0.03	0.00	0.00	0.00	0.00	0.00	-0.01	0.00	0.00	0.02
	10	2		0.08	0.08	0.12	0.01	0.02	0.01	0.01	0.01	0.01	0.02	0.02	0.01	-0.01	-0.01	0.00
		3		-0.01	-0.01	0.00	-0.03	-0.02	-0.03	-0.01	0.00	-0.01	-0.02	-0.03	-0.03	0.03	-0.04	0.00
		4		-0.03	-0.02	-0.01	-0.03	-0.01	-0.02	-0.01	-0.02	-0.01	-0.02	-0.02	-0.02	0.00	0.00	-0.01
3	4	2		-0.01	0.00	-0.01	0.04	0.06	0.01	0.04	0.29	0.29	0.02	0.14	0.29	0.02	0.29	0.29
		3		0.00	0.02	0.02	-0.01	-0.01	0.00	0.01	0.08	0.05	0.00	0.00	0.08	0.07	0.04	0.04
		4		0.00	0.01	0.00	-0.03	-0.02	-0.01	-0.01	0.03	0.02	-0.01	-0.02	0.04	0.04	0.08	0.02
	6	2		-0.01	0.00	-0.01	0.29	0.06	0.01	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29
		3		0.00	0.02	0.03	0.11	0.00	0.00	0.11	0.08	0.05	0.08	0.05	0.07	0.07	0.04	0.04
		4		0.00	0.01	0.00	0.03	-0.02	0.00	0.05	0.03	0.02	0.04	0.03	0.04	0.04	0.08	0.02
	8	2		-0.01	0.00	-0.01	0.29	0.06	0.01	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29
		3		0.00	0.02	0.03	0.11	0.00	0.00	0.11	0.08	0.06	0.08	0.05	0.07	0.07	0.04	0.04
		4		0.00	0.01	0.00	0.03	-0.02	0.00	0.05	0.03	0.00	0.04	0.03	0.04	0.05	0.09	0.02
	10	2		0.02	0.02	0.00	0.29	0.00	0.02	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29
		3		0.04	0.00	0.01	0.12	0.03	0.00	0.09	0.10	0.06	0.06	0.05	0.06	0.04	0.05	0.05
		4		0.00	0.00	0.00	0.03	0.01	-0.01	0.04	0.01	0.01	0.01	0.01	0.04	0.03	0.03	0.06
4	4	2		0.01	0.06	0.01	0.00	-0.01	-0.01	-0.01	0.03	-0.03	0.01	0.00	0.00	-0.02	0.29	0.29
		3		0.02	0.01	-0.02	-0.02	0.01	0.00	-0.01	0.02	0.00	-0.01	-0.02	0.05	0.01	0.04	0.05
		4		-0.01	0.00	-0.03	-0.01	-0.01	-0.02	0.01	0.02	0.02	0.00	0.01	0.02	0.03	0.02	0.03
	6	2		0.01	0.07	0.01	0.00	-0.02	-0.01	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29
		3		0.06	0.01	-0.02	-0.02	0.01	-0.01	0.04	0.04	0.04	0.07	0.05	0.05	0.04	0.04	0.05
		4		0.04	0.00	-0.03	0.01	-0.01	-0.02	0.03	0.04	0.03	0.03	0.01	0.02	0.05	0.03	0.03
	8	2		0.29	0.02	0.29	0.29	-0.02	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29
		3		0.06	0.01	0.06	0.05	0.01	0.04	0.06	0.05	0.04	0.07	0.05	0.05	0.04	0.04	0.05
		4		0.04	0.00	0.04	0.02	-0.01	0.03	0.03	0.05	0.03	0.03	0.01	0.03	0.05	0.03	0.03
	10	2		0.29	-0.01	0.29	0.29	0.01	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29
		3		0.08	0.01	0.10	0.04	0.02	0.04	0.06	0.04	0.06	0.05	0.05	0.10	0.15	0.04	0.06
		4		0.04	0.04	0.02	0.06	0.02	0.04	0.04	0.02	0.03	0.01	0.04	0.05	0.03	0.01	0.02

In HMM MSTC on Arçelik data, increasing the number of hidden states makes it possible to keep more detail about system characteristics in the model. As expected, higher number of iterations ($W \geq 8$) allow more accurate modelling with higher levels of convergence. A combination of high number of states and high number of iterations also allows the method

to be robust to the different values of other parameters.

The failure of all the methods to achieve a perfect clustering of MTS on these data sets may be due to several reasons. Calibrating the Lego Demonstrator components are challenging. Bad calibrations can result in change of the applied force by the motors and/or the time of accomplishing specific actions. Such changes might not be visible with eye but can affect the complete cycle of the demonstrator and the corresponding recorded MTS. In Arçelik Data, the assumption of "the manifestation of the anomalies only occurs in the MTS that is recorded just before the anomaly" (see 6.2) might be inaccurate and the deviation from the normal behavior might be happening much earlier. Considering these factors, a perfectly accurate clustering might be even impossible with the available setup and data sets.

Table 7: Adjusted Rand Index Results of VOMM MTS Clustering done with the combinations of parameters explained in table 2 on Arcelik Data. Adjusted RAND Index (ARI) is used for scoring.

			a=8									a=9									a=10									
			d=4			d=5			d=6			d=4			d=5			d=6			d=4			d=5			d=6			
t	L	mk	vk=5	6	7	5	6	7	5	6	7	5	6	7	5	6	7	5	6	7	5	6	7	5	6	7	5	6	7	
4	5	3	0.3	-0.1	0.1	0.1	0.0	0.1	0.2	0.1	0.1	0.1	0.1	0.0	0.2	0.2	0.1	0.2	0.0	0.1	0.2	0.1	-0.1	0.2	-0.1	-0.1	0.1	0.1	-0.1	
		4	0.2	0.0	0.1	0.1	0.0	0.0	0.2	0.2	0.1	0.0	0.1	0.0	0.1	0.2	0.3	0.1	0.1	0.0	0.1	0.3	0.0	-0.1	0.2	0.1	0.1	0.1	0.1	0.1
		5	0.2	0.0	0.1	0.0	0.0	0.0	0.3	0.3	0.0	0.0	0.1	0.0	0.2	0.3	0.1	0.0	0.0	0.1	0.2	0.0	-0.1	0.3	0.1	0.1	0.1	0.1	0.1	0.0
	6	3	0.53	0.1	0.2	0.1	0.4	0.1	0.1	0.4	0.1	0.1	0.2	0.1	0.2	0.1	0.1	0.1	0.0	0.1	0.44	0.1	0.0	0.4	0.2	0.61	0.0	0.2	0.2	
		4	0.4	0.1	0.2	0.0	0.3	0.3	0.1	0.5	0.0	0.1	0.2	0.1	0.2	0.1	0.1	0.0	0.0	0.0	0.4	0.1	0.0	0.3	0.3	0.53	0.0	0.2	0.3	
		5	0.4	0.0	0.3	0.0	0.3	0.3	0.1	0.4	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.0	0.1	0.1	0.3	0.2	0.0	0.3	0.3	0.47	0.0	0.3	0.3	
	7	3	0.4	0.1	0.1	0.0	0.4	0.4	0.1	0.4	0.0	0.1	0.2	-0.1	0.2	0.1	0.2	0.0	0.1	0.1	0.3	0.1	-0.1	0.4	0.3	0.4	-0.1	0.4	0.4	
		4	0.3	0.2	0.2	0.0	0.3	0.3	0.1	0.5	0.0	0.0	0.2	0.1	0.2	0.2	0.2	0.0	0.0	0.0	0.2	0.2	-0.1	0.3	0.3	0.3	0.0	0.3	0.3	
		5	0.3	0.2	0.2	0.1	0.3	0.3	0.1	0.4	0.2	0.0	0.1	0.2	0.1	0.2	0.1	0.0	0.1	0.0	0.2	0.1	-0.1	0.3	0.2	0.3	0.0	0.3	0.3	
5	5	3	0.3	0.1	0.1	0.1	0.1	0.1	0.1	0.3	0.1	0.2	0.1	0.1	0.1	0.2	0.1	0.2	0.0	0.1	0.3	0.0	0.1	0.4	0.1	0.2	0.1	0.2	0.2	
		4	0.4	0.0	0.1	0.0	0.1	0.3	0.3	0.4	0.0	0.1	0.1	0.1	0.2	0.2	0.1	0.1	0.0	0.1	0.3	0.0	0.0	0.3	0.1	0.3	0.0	0.2	0.1	
		5	0.4	0.1	0.1	0.0	0.3	0.2	0.4	0.4	0.0	0.1	0.1	0.0	0.4	0.2	0.1	0.0	0.0	0.0	0.2	0.0	0.0	0.2	0.1	0.3	0.0	0.1	0.1	
	6	3	0.4	0.2	0.1	0.0	0.1	0.1	0.1	0.53	0.1	0.1	0.1	-0.1	0.2	0.2	0.1	0.1	0.0	0.1	0.2	0.2	0.1	0.1	0.4	0.3	0.0	0.4	0.1	
		4	0.3	0.1	0.3	0.0	0.1	0.1	0.1	0.4	0.0	0.1	0.1	-0.1	0.2	0.4	0.1	0.0	0.0	0.1	0.2	0.3	0.0	0.2	0.3	0.4	0.0	0.3	0.1	
		5	0.4	0.2	0.1	0.0	0.1	0.1	0.1	0.4	0.1	0.0	0.1	0.1	0.1	0.3	0.1	0.0	0.0	0.0	0.2	0.2	0.0	0.2	0.3	0.4	0.0	0.2	0.1	
	7	3	0.1	0.1	0.1	0.0	0.1	0.1	0.1	0.3	0.2	0.0	0.0	-0.1	0.3	0.2	0.0	0.1	0.0	0.1	0.2	0.3	0.1	0.1	0.1	0.2	0.0	0.1	0.2	
		4	0.1	0.1	0.3	0.0	0.1	0.1	0.1	0.4	0.4	0.0	0.2	-0.1	0.3	0.3	0.2	0.0	0.0	0.0	0.2	0.3	0.1	0.2	0.2	0.3	0.0	0.2	0.3	
		5	0.1	0.1	0.2	0.0	0.1	0.1	0.1	0.4	0.3	0.0	0.1	0.1	0.2	0.3	0.1	0.0	0.1	0.0	0.2	0.2	0.1	0.2	0.2	0.3	0.0	0.2	0.3	
6	5	3	0.2	0.2	0.0	0.1	0.4	0.5	0.2	0.3	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.3	0.1	0.0	0.1	-0.1	-0.1	0.0	-0.1	-0.1	
		4	0.4	0.1	0.0	0.0	0.3	0.4	0.1	0.4	0.0	0.0	0.1	0.1	0.1	0.2	0.1	0.1	0.0	0.1	0.5	0.1	0.0	0.2	-0.1	0.0	0.0	-0.1	0.0	
		5	0.3	0.1	0.0	0.0	0.3	0.4	0.2	0.4	0.0	0.0	0.1	0.1	0.1	0.3	0.1	0.0	0.0	0.0	0.4	0.2	0.0	0.2	0.1	0.0	0.0	0.1	0.3	
	6	3	0.4	0.2	0.0	0.0	0.1	0.3	0.2	0.3	0.2	0.1	0.1	0.2	0.5	0.2	0.1	0.0	0.1	0.1	0.2	0.3	0.0	0.1	0.1	0.0	0.0	0.1	0.1	
		4	0.3	0.1	0.1	0.0	0.1	0.3	0.3	0.4	0.2	0.0	0.1	0.2	0.4	0.4	0.1	0.0	0.1	0.0	0.3	0.3	0.0	0.2	0.0	0.0	0.0	0.1	0.2	
		5	0.3	0.1	0.2	0.0	0.1	0.2	0.3	0.4	0.1	0.0	0.1	0.1	0.4	0.3	0.1	0.0	0.1	0.0	0.3	0.2	0.0	0.2	0.2	0.1	0.0	0.2	0.2	
	7	3	0.1	0.1	0.1	0.0	0.2	0.2	0.2	0.2	0.1	0.1	0.1	0.1	0.2	0.2	0.0	0.0	0.1	0.0	0.1	0.3	0.0	0.1	0.1	0.1	0.0	0.1	0.1	
		4	0.1	0.1	0.2	0.0	0.3	0.3	0.3	0.3	0.1	0.0	0.1	0.1	0.3	0.3	0.2	0.0	0.0	0.0	0.2	0.2	0.0	0.2	0.0	0.2	0.0	0.0	0.2	
		5	0.1	0.1	0.2	0.1	0.3	0.3	0.3	0.3	0.2	0.0	0.1	0.1	0.3	0.3	0.1	0.0	0.0	0.0	0.2	0.2	0.0	0.2	0.2	0.1	0.0	0.1	0.2	

The poor performance of PCA MTSC shows the requirement for a more powerful method such as Markov Models, which can capture temporal dependencies of internal states of the system. Despite the high complexity of HMM MTSC method, it is outperformed by VOMM MTSC

method, which also has better performance in the manner of time complexity. This can be expected since that VOMMs can fit better to the variations of the length of the patterns in the data and such variations occur with a high frequency in the data from industrial area. The drawback of VOMM MTSC is the high number of parameters; the desired settings for clustering parameters can be estimated by the clustering evaluation methods up to some degree.



7 CONCLUSION

The challenges in collecting labelled data in the industrial systems is one of the important issues in tasks such as anomaly detection or behavior identification applied on these systems. Therefore, detection of the system modes from the sensory data which lacks human expert labels is a real-world example of the well-studied Multivariate Time Series Clustering problem. We proposed a novel method and compared it with the two existing MTS Clustering methods and confirmed the superiority of the proposed method on two industrial data sets. Success on distinguishing normal and anomalous modes of the system is also tested by using one data set which is recorded in a period where the system went through several normal and anomalous modes. We also presented an extensive complexity analysis and comparison of these three methods.

Expert knowledge can help for the setting of hyper-parameters in VOMM MTS Clustering up to some degree. L should be set to a small number preferably $L < 10$, so while significant information is still represented in the model and time and space complexity is not overwhelmed. While PCA has some contribution in the success, the method is robust to parameter d in experiments showed in general a setting that will result in 90% explained variance is appropriate. One should set a considering the record frequency of the sensor data, high frequency requires high values of a . One approach to set mk and k is using *clustering validation* techniques as presented in Appendix A. The experiments on Demonstrator data with clustering validation gave promising results while in Arçelik data it gave worse results, this introduces the requirement of further study in this sub-task. Parameters t and I should be adjusted experimentally with the existing labelled data. An appropriate choice of t brings robustness to the method against parameter a .

Table 8: Adjusted Rand Index Results of VOMM MTS Clustering done with the combinations of parameters explained in table 2 on Demonstrator Data. Adjusted RAND Index (ARI) is used for scoring.

t	L	mk	a= d=	6									7									8														
				7			10			15			7			10			15			7			10			15								
				vk=	2	3	4	2	3	4	2	3	4	2	3	4	2	3	4	2	3	4	2	3	4	2	3	4	2	3	4	2	3	4		
3	5	2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.1	0.2	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1
		3	0.2	0.3	0.0	0.2	0.1	0.0	0.2	0.0	0.0	0.3	0.0	0.0	0.3	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.0	0.1	0.1	0.0	0.0	0.1	0.1	0.1	0.1	0.0	0.0	0.1	0.0	0.1
		4	0.2	0.3	0.0	0.2	0.3	0.0	0.2	0.0	0.0	0.4	0.4	0.0	0.3	0.3	0.1	0.3	0.0	0.0	0.1	0.5	0.0	0.0	0.4	0.0	0.1	0.0	0.0	0.4	0.0	0.1	0.0	0.0	0.0	0.0
		5	0.1	0.3	0.0	0.1	0.4	0.0	0.1	0.3	0.0	0.3	0.4	0.0	0.3	0.2	0.1	0.3	0.0	0.0	0.1	0.5	0.0	0.1	0.5	0.1	0.0	0.1	0.5	0.1	0.0	0.4	0.1	0.0	0.1	
		6	2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.1	0.2	0.0	0.0	0.4	0.0	0.0	0.1	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.1	0.1	0.1	0.1	
	7	3	0.3	0.3	0.0	0.3	0.1	0.0	0.3	0.0	0.0	0.3	0.0	0.0	0.3	0.0	0.0	0.4	0.0	0.1	0.1	0.1	0.0	0.0	0.1	0.0	0.0	0.1	0.0	0.1	0.0	0.1	0.0	0.1		
		4	0.3	0.3	0.0	0.3	0.4	0.0	0.3	0.0	0.1	0.4	0.1	0.0	0.2	0.1	0.1	0.4	0.0	0.1	0.1	0.1	0.0	0.0	0.4	0.0	0.1	0.0	0.0	0.1	0.0	0.1	0.0	0.1		
		5	0.4	0.4	0.0	0.4	0.5	0.1	0.4	0.1	0.1	0.3	0.5	0.0	0.3	0.5	0.1	0.3	0.0	0.1	0.1	0.2	0.0	0.1	0.4	0.0	0.1	0.2	0.0	0.1	0.2	0.1	0.2	0.1		
		2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.2	0.0	0.0	0.6	0.0	0.0	0.1	0.0	0.0	0.2	0.0	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1		
	4	5	2	0.5	0.1	0.0	0.5	0.1	0.0	0.5	0.0	0.0	0.0	0.0	0.2	0.0	0.1	0.5	0.0	0.0	0.1	0.1	0.0	0.2	0.1	0.0	0.2	0.1	0.0	0.2	0.1	0.0	0.1	0.1		
			3	0.5	0.5	0.0	0.5	0.4	0.1	0.5	0.0	0.0	0.3	0.1	0.0	0.2	0.1	0.1	0.3	0.0	0.2	0.1	0.1	0.3	0.0	0.2	0.1	0.1	0.0	0.2	0.1	0.0	0.2	0.1	0.0	
			4	0.4	0.5	0.0	0.4	0.4	0.1	0.4	0.1	0.1	0.3	0.2	0.0	0.3	0.3	0.1	0.3	0.0	0.1	0.1	0.1	0.0	0.3	0.0	0.1	0.1	0.1	0.0	0.3	0.0	0.0	0.2	0.1	0.1
			5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.2	0.0	0.0	0.6	0.0	0.0	0.1	0.0	0.0	0.2	0.0	0.0	0.1	0.0	0.0	0.1	0.1	0.1	0.1	0.1	
			6	2	0.2	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.1	0.2	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
		5	6	3	0.2	0.3	0.0	0.2	0.1	0.0	0.2	0.0	0.1	0.3	0.0	0.0	0.3	0.0	0.0	0.2	0.0	0.0	0.1	0.1	0.0	0.0	0.1	0.0	0.0	0.1	0.0	0.1	0.0	0.1	0.0	0.1
4				0.2	0.3	0.0	0.2	0.3	0.0	0.2	0.0	0.0	0.4	0.4	0.1	0.3	0.3	0.0	0.3	0.0	0.0	0.1	0.5	0.0	0.0	0.4	0.0	0.1	0.0	0.0	0.4	0.0	0.1	0.0	0.0	
5				0.3	0.3	0.0	0.3	0.4	0.0	0.3	0.3	0.0	0.3	0.4	0.1	0.3	0.2	0.0	0.3	0.0	0.1	0.1	0.5	0.0	0.1	0.5	0.0	0.1	0.5	0.0	0.0	0.4	0.0	0.0	0.4	0.0
2				0.3	0.0	0.0	0.3	0.0	0.0	0.3	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.0	0.4	0.0	0.0	0.1	0.1	0.0	0.0	0.0	0.1	0.1	0.0	0.0	0.0	0.0	0.1	0.1	0.0	
7			3	0.3	0.3	0.0	0.3	0.1	0.0	0.3	0.0	0.1	0.3	0.0	0.0	0.3	0.0	0.0	0.4	0.0	0.0	0.1	0.1	0.0	0.0	0.1	0.0	0.0	0.1	0.0	0.1	0.0	0.1	0.0	0.1	
			4	0.4	0.3	0.0	0.4	0.4	0.0	0.4	0.0	0.0	0.4	0.1	0.1	0.2	0.1	0.0	0.4	0.0	0.0	0.1	0.1	0.0	0.0	0.4	0.0	0.1	0.0	0.0	0.4	0.0	0.1	0.0	0.0	
			5	0.3	0.4	0.0	0.3	0.5	0.0	0.3	0.1	0.0	0.3	0.5	0.1	0.3	0.5	0.1	0.3	0.0	0.0	0.1	0.2	0.0	0.1	0.4	0.1	0.1	0.2	0.0	0.1	0.2	0.0	0.2	0.0	
			2	0.5	0.0	0.0	0.5	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.1	0.2	0.0	0.0	0.6	0.0	0.0	0.1	0.0	0.0	0.2	0.0	0.0	0.1	0.0	0.0	0.1	0.1	0.1	0.1	0.0	
			3	0.5	0.1	0.0	0.5	0.1	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.5	0.0	0.0	0.1	0.1	0.0	0.2	0.0	0.0	0.1	0.1	0.0	0.2	0.0	0.0	0.1	0.1	0.1
			4	0.4	0.5	0.0	0.4	0.4	0.0	0.4	0.0	0.0	0.3	0.1	0.1	0.2	0.1	0.0	0.3	0.0	0.0	0.1	0.1	0.0	0.2	0.1	0.0	0.2	0.1	0.0	0.2	0.1	0.0	0.2	0.1	0.0
5	5	2	0.3	0.5	0.0	0.3	0.4	0.0	0.3	0.1	0.0	0.3	0.2	0.1	0.3	0.3	0.1	0.3	0.0	0.0	0.1	0.1	0.0	0.3	0.1	0.1	0.2	0.1	0.0	0.3	0.1	0.1	0.2	0.1	0.0	
		3	0.2	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
		4	0.2	0.3	0.0	0.2	0.3	0.0	0.2	0.0	0.0	0.4	0.4	0.0	0.3	0.3	0.0	0.3	0.0	0.0	0.1	0.5	0.0	0.0	0.4	0.1	0.1	0.0	0.0	0.4	0.1	0.1	0.0	0.0	0.0	
		5	0.3	0.3	0.0	0.3	0.4	0.0	0.3	0.3	0.0	0.3	0.4	0.0	0.3	0.2	0.0	0.3	0.0	0.0	0.1	0.5	0.0	0.1	0.5	0.1	0.0	0.1	0.5	0.1	0.0	0.4	0.0	0.0	0.4	0.0
		6	2	0.3	0.0	0.0	0.3	0.0	0.0	0.3	0.0	0.0	0.2	0.0	0.0	0.2	0.0	0.0	0.4	0.0	0.0	0.1	0.1	0.0	0.0	0.0	0.0	0.1	0.1	0.0	0.0	0.0	0.1	0.0	0.0	
	6	6	3	0.3	0.3	0.0	0.3	0.1	0.0	0.3	0.0	0.0	0.3	0.0	0.0	0.3	0.0	0.0	0.4	0.0	0.1	0.1	0.0	0.0	0.1	0.1	0.0	0.0	0.1	0.1	0.1	0.0	0.0	0.0	0.0	
			4	0.4	0.3	0.0	0.4	0.4	0.0	0.4	0.0	0.0	0.4	0.1	0.0	0.2	0.1	0.0	0.4	0.0	0.1	0.1	0.0	0.0	0.4	0.0	0.1	0.1	0.0	0.0	0.4	0.1	0.1	0.0	0.0	
			5	0.3	0.4	0.0	0.3	0.5	0.0	0.3	0.1	0.0	0.3	0.5	0.0	0.3	0.5	0.0	0.3	0.0	0.1	0.1	0.2	0.0	0.1	0.4	0.1	0.1	0.2	0.0	0.1	0.2	0.0	0.2	0.0	
			2	0.5	0.0	0.0	0.5	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.6	0.0	0.0	0.1	0.0	0.0	0.2	0.0	0.0	0.1	0.0	0.0	0.2	0.0	0.1	0.1	0.0	0.0
		7	3	0.5	0.1	0.0	0.5	0.1	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.5	0.0	0.1	0.1	0.0	0.2	0.0	0.1	0.1	0.0	0.2	0.0	0.1	0.1	0.1	0.1	0.1	
			4	0.4	0.5	0.0	0.4	0.4	0.0	0.4	0.0	0.0	0.3	0.1	0.0	0.2	0.1	0.0	0.3	0.0	0.1	0.1	0.1	0.0	0.3	0.0	0.1	0.1	0.1	0.0	0.3	0.1	0.1	0.2	0.1	0.0
			5	0.3	0.5	0.0	0.3	0.4	0.0	0.3	0.1	0.0	0.3	0.2	0.0	0.3	0.3	0.0	0.3	0.0	0.1	0.1	0.1	0.0	0.3	0.1	0.1	0.1	0.0	0.3	0.1	0.1	0.2	0.1	0.0	0.0
			2	0.2	0.0	0.1	0.2	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.2	0.1	0.2	0.1	0.0	0.0	0.1	0.1	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.0	0.0	0.0
			3	0.2	0.1	0.1	0.2	0.1	0.0	0.2	0.1	0.0	0.0	0.1	0.1	0.0	0.1	0.0	0.2	0.1	0.1	0.1	0.1	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.0
			4	0.2	0.1	0.1	0.2	0.1	0.1	0.2	0.0	0.1	0.0	0.2	0.1	0.0	0.1	0.1	0.3	0.2	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.0
6	6	2	0.3	0.1	0.1	0.3	0.1	0.0	0.3	0.1	0.0	0.5	0.3	0.1	0.4	0.0	0.0	0.4	0.2	0.1	0.3	0.2	0.0	0.4	0.2	0.1	0.3	0.2	0.0	0.4	0.2	0.1	0.3	0.1	0.0	
		3	0.3	0.2	0.1	0.3	0.2	0.1	0.3	0.1	0.0	0.3	0.2	0.2	0.2	0.1	0.2	0.4	0.1	0.1	0.3	0.3	0.1	0.4	0.3	0.2	0.3	0.3	0.1	0.4	0.3	0.2	0.3	0.3	0.1	
		4	0.4	0.2	0.1	0.4	0.2	0.1	0.4	0.2	0.1	0.4	0.2	0.1	0.1	0.2	0.2	0.4	0.3	0.1	0.3	0.2	0.1	0.3	0.4	0.1	0.3	0.2	0.1	0.3	0.4	0.1	0.3	0.3	0.1	
		5	0.3	0.2	0.1	0.3	0.2	0.1	0.3																											

We conclude that VOMM MTS Clustering is bound to errors, but it is still significantly successful in identifying similarities and differences of the characteristics of the MTS in the applications where the labeled data are hard to obtain. This is experimentally shown on one physical demonstrator data and one real world industrial system data. Learned clusters with the proposed method can be used as preliminary information/models for dealing with important tasks such as anomaly detection or root-cause analysis in industrial systems.



8 REFERENCES

- [1] Mell, Peter, and Tim Grance. The NIST definition of cloud computing. (2011).
- [2] Steuer, Jonathan. Defining virtual reality: Dimensions determining telepresence. *Journal of communication* 42.4 (1992) 73-93.
- [3] Gubbi, Jayavardhana, et al. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future generation computer systems* 29.7 (2013) 1645-1660.
- [4] Lasi, Heiner, et al. Industry 4.0. *Business & information systems engineering* 6.4 (2014) 239-242.
- [5] G. E. Box, G. M. Jenkins, G. C. Reinsel, G. M. Ljung, *Time series analysis: forecasting and control*, John Wiley & Sons, 2015.
- [6] E. Alpaydin, *Introduction to Machine Learning*, MIT Press, 2014.
- [7] Pappas, Thrasyvoulos N. An adaptive clustering algorithm for image segmentation. *IEEE Transactions on signal processing* 40.4 (1992) 901-914.
- [8] He, Zengyou, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. *Pattern Recognition Letters* 24.9-10 (2003) 1641-1650.
- [9] Tichy, Noel M., Michael L. Tushman, and Charles Fombrun. Social network analysis for organizations. *Academy of management review* 4.4 (1979) 507-519.
- [10] B. G. Sürmeli, F. Eksen, B. Dinç, P. Schüller, B. Tümer, Unsupervised mode detection in cyber-physical systems using variable order markov models, in: *Industrial Informatics (INDIN)*, 2017 IEEE 15th International Conference on, IEEE, 2017, pp. 841–846.
- [11] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection for discrete sequences: A survey, *IEEE Transactions on Knowledge and Data Engineering* 24 (5) (2012) 823–839.
- [12] O. Niggemann, B. Stein, A. Vodencarevic, A. Maier, H. K. Büning, Learning behavior

models for hybrid timed systems., in: AAAI, 2012, pp. 1083–1090.

- [13] O. Niggemann, S. Windmann, S. Volgmann, A. Bunte, B. Stein, Using learned models for the root cause analysis of cyber-physical production systems, in: Int. Workshop Principles of Diagnosis (DX), 2014.
- [14] T. W. Liao, A clustering procedure for exploratory mining of vector time series, *Pattern Recognition* 40 (9) (2007) 2550–2562.
- [15] Y. Kakizawa, R. H. Shumway, M. Taniguchi, Discrimination and clustering for multivariate time series, *Journal of the American Statistical Association* 93 (441) (1998) 328–340.
- [16] K. Kosmelj, V. Batagelj, Cross-sectional approach for clustering time varying data, *Journal of Classification* 7 (1) (1990) 99–109.
- [17] Singhal, D. E. Seborg, Clustering multivariate time-series data, *Journal of Chemometrics: A Journal of the Chemometrics Society* 19 (8) (2005) 427–438.
- [18] S. Ghassempour, F. Girosi, A. Maeder, Clustering multivariate time series using hidden markov models, *International journal of environmental research and public health* 11 (3) (2014) 2741–2763.
- [19] P. Bühlmann, A. J. Wyner, et al., Variable length markov chains, *The Annals of Statistics* 27 (2) (1999) 480–513.
- [20] Arçelik A.Ş., [Online; accessed 13-January-2019] URL: <http://www.arcelikas.com/>.
- [21] Papoulis, Athanasios, and S. Unnikrishna Pillai. *Probability, random variables, and stochastic processes*. Tata McGraw-Hill Education, (2002).
- [22] G. Bejerano, G. Yona, Variations on probabilistic suffix trees: statistical modeling and prediction of protein families, *Bioinformatics* 17 (1) (2001) 23–43.

- [23] M. H. Schulz, D. Weese, T. Rausch, A. Döring, K. Reinert, M. Vingron, Fast and adaptive variable order markov chain construction, in: International Workshop on Algorithms in Bioinformatics, 2008, pp. 306–317.
- [24] D. Ron, Y. Singer, N. Tishby, Learning probabilistic automata with variable memory length, in: Proc. Computational Learning Theory, 1994, pp. 35–46.
- [25] J. Rissanen, A universal data compression system, IEEE Transactions on Information Theory 29 (5) (1983) 656–664.
- [26] P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the em algorithm, Journal of the royal statistical society. Series B (methodological) (1977) 1–38.
- [27] J. D. Hamilton, Time series analysis, Vol. 2, Princeton university press Princeton, NJ, 1994.
- [28] K. Wagner, S. B. Soumerai, F. Zhang, D. Ross-Degnan, Segmented regression analysis of interrupted time series studies in medication use research, Journal of clinical pharmacy and therapeutics 27 (4) (2002) 299–309.
- [29] C.-K. Peng, S. Havlin, H. E. Stanley, A. L. Goldberger, Quantification of scaling exponents and crossover phenomena in nonstationary heartbeat time series, Chaos: An Interdisciplinary Journal of Nonlinear Science 5 (1) (1995) 82–87.
- [30] Z. Bar-Joseph, Analyzing time series gene expression data, Bioinformatics 20 (16) (2004) 2493–2503.
- [31] J. Aach, G. M. Church, Aligning gene expression time series with time warping algorithms, Bioinformatics 17 (6) (2001) 495–508.
- [32] Tumer, M. B., Lee A. Belfore, and Kristina M. Ropella. "A syntactic methodology for automatic diagnosis by analysis of continuous time measurements using hierarchical signal representations." IEEE Transactions on Systems, Man, and Cybernetics, Part B

(Cybernetics) 33.6 (2003): 951-965.

- [33] Brockwell, Peter J., Richard A. Davis, and Matthew V. Calder. Introduction to time series and forecasting. Vol. 2. New York: springer, 2002.
- [34] Bontempi, Gianluca, Souhaib Ben Taieb, and Yann-Aël Le Borgne. "Machine learning strategies for time series forecasting." European business intelligence summer school. Springer, Berlin, Heidelberg, 2012.
- [35] T. W. Liao, Clustering of time series data—a survey, *Pattern recognition* 38 (11)(2005) 1857–1874.
- [36] S. Aghabozorgi, A. S. Shirkhorshidi, T. Y. Wah, Time-series clustering—a decade review, *Information Systems* 53 (2015) 16–38.
- [37] 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*, ACM, 2003, pp. 2–11.
- [38] E. S. Dimitrova, M. P. V. Licona, J. McGee, R. Laubenbacher, Discretization of time series data, *Journal of Computational Biology* 17 (6) (2010) 853–868.
- [39] L. M. Owsley, L. E. Atlas, G. D. Bernard, Self-organizing feature maps and hidden markov models for machine-tool monitoring, *IEEE Transactions on Signal Processing* 45 (11) (1997) 2787–2798.
- [40] M. G. Baydogan, G. Runger, Learning a symbolic representation for multivariate time series classification, *Data Mining and Knowledge Discovery* 29 (2) (2015) 400–422.
- [41] G. Das, K.-I. Lin, H. Mannila, G. Renganathan, P. Smyth, Rule discovery from time series., in: *KDD*, Vol. 98, 1998, pp. 16–22.
- [42] von Birgelen, O. Niggemann, Enable learning of hybrid timed automata in absence of discrete events through self-organizing maps, in: *IMPROVE-Innovative Modelling*

Approaches for Production Systems to Raise Validatable Efficiency, Springer, 2018, pp. 37–54.

- [43] R. Moskovitch, Y. Shahar, Classification-driven temporal discretization of multivariate time series, *Data Mining and Knowledge Discovery* 29 (4) (2015) 871–913.
- [44] R. Begleiter, R. El-Yaniv, G. Yona, On prediction using variable order markov models, *Journal of Artificial Intelligence Research* 22 (2004) 385–421.
- [45] H. Oğul, E. Ü. Mumcuoğlu, SVM-based detection of distant protein structural relationships using pairwise probabilistic suffix trees, *Computational Biology and Chemistry* 30 (4) (2006) 292–299.
- [46] Apostolico, G. Bejerano, Optimal amnesic probabilistic automata or how to learn and classify proteins in linear time and space, *Journal of Computational Biology* 7 (3-4) (2000) 381–393.
- [47] P. Sun, S. Chawla, B. Arunasalam, Mining for outliers in sequential databases, in: *Proc. SIAM International Conference on Data Mining*, 2006, pp. 94–105.
- [48] Aberg, J., and Yu M. Shtarkov. "Text compression by context tree weighting." *Proceedings DCC'97. Data Compression Conference. IEEE*, 1997.
- [49] Sadakane, Kunihiko, Takumi Okazaki, and Hiroshi Imai. "Implementing the context tree weighting method for text compression." *Proceedings DCC 2000. Data Compression Conference. IEEE*, 2000.
- [50] Witten, Ian H., and Timothy C. Bell. "The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression." *Ieee transactions on information theory* 37.4 (1991): 1085-1094.
- [51] Pachet, François. "Playing with virtual musicians: The continuator in practice." *IEEE MultiMedia* 9.3 (2002): 77-82.

- [52] S. R. Eddy, Hidden markov models, *Current opinion in structural biology* 6 (3) (1996) 361–365.
- [53] O. Niggemann, A. Vodencarevic, A. Maier, S. Windmann, H. K. Büning, A learning anomaly detection algorithm for hybrid manufacturing systems, in: *International Workshop on Principles of Diagnosis*, 2013.
- [54] P. Wunderlich, O. Niggemann, Structure learning methods for bayesian networks to reduce alarm floods by identifying the root cause, in: *Emerging Technologies and Factory Automation (ETFA), 2017 22nd IEEE International Conference on*, IEEE, 2017, pp. 1–8.
- [55] H. Chim, X. Deng, A new suffix tree similarity measure for document clustering, in: *Proc. WWW*, 2007, pp. 121–130.
- [56] F. Murtagh, P. Legendre, Ward’s hierarchical agglomerative clustering method: which algorithms implement ward’s criterion?, *Journal of classification* 31 (3) (2014) 274–295.
- [57] R. Giegerich, S. Kurtz, J. Stoye, Efficient implementation of lazy suffix trees, *Software: Practice and Experience* 33 (11) (2003) 1035–1049.
- [58] Ukkonen, Esko. On-line construction of suffix trees. *Algorithmica* 14.3 (1995): 249-260.
- [59] Cormen, Thomas H., et al. 8.2: Counting sort. *Introduction to Algorithms*, (2001): 168-170.
- [60] J. Gustafsson, E. Norlander, Clustering genomic signatures. Master’s Thesis, Department of Computer Science and Engineering Chalmers University of Technology, University of Gothenburg, Gothenburg, Sweden (2018).
- [61] F. Cuzzolin, M. Sapienza, Learning pullback hmm distances, *IEEE transactions on pattern analysis and machine intelligence* 36 (7) (2014) 1483–1489.

- [62] C. Li, G. Biswas, Temporal pattern generation using hidden markov model based unsupervised classification, in: *International Symposium on Intelligent Data Analysis*, Springer, 1999, pp. 245–256.
- [63] G. Pfundstein, Hidden markov models with generalised emission distribution for the analysis of high-dimensional, non-euclidean data, Ph.D. thesis, Institut für Statistik (2011).
- [64] M. N. Do, M. Vetterli, Rotation invariant texture characterization and retrieval using steerable wavelet-domain hidden markov models, *IEEE Transactions on Multimedia* 4 (LCAV-ARTICLE-2002-002) (2002) 517–527.
- [65] R. J. Campello, D. Moulavi, A. Zimek, J. Sander, Hierarchical density estimates for data clustering, visualization, and outlier detection, *ACM Transactions on Knowledge Discovery from Data (TKDD)* 10 (1) (2015) 5.
- [66] L. Kaufman, P.J. Rousseeuw, Partitioning around medoids, in: *Finding groups in data: an introduction to cluster analysis*, Wiley Online Library, 1990.
- [67] Sharma, K. K. Paliwal, Fast principal component analysis using fixed-point algorithm, *Pattern Recognition Letters* 28 (10) (2007) 1151–1155.
- [68] M. Holmes, A. Gray, C. Isbell, Fast svd for large-scale matrices, in: *Workshop on Efficient Machine Learning at NIPS*, Vol. 58, 2007, pp. 249–252.
- [69] Vinh, Nguyen Xuan, Julien Epps, and James Bailey. "Information theoretic measures for clusterings comparison: is a correction for chance necessary?." *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009.
- [70] M. Charrad, N. Ghazzali, V. Boiteau, A. Niknafs, M. M. Charrad, Package ‘nbclust’, *Journal of Statistical Software* 61 (2014) 1–36.
- [71] T. Frey, H. Van Groenewoud, A cluster analysis of the d2 matrix of white spruce stands in saskatchewan based on the maximum-minimum principle, *The Journal of Ecology*

(1972) 873–886.

- [72] Kawaguchi, Nobuo, et al. "HASC Challenge: gathering large scale human activity corpus for the real-world activity understandings." Proceedings of the 2nd augmented human international conference. ACM, 2011.
- [73] HASC Challenge 2014, [Online; Last Accessed 30-July-2019], <http://hasc.jp/hc2014/dataFormat-en.html>



Appendix A. Determining the Number of Clusters in VOMM MTS Clustering

Estimating the real number of clusters in the data is an important challenge. Since in clustering problem, we do not have a labeled data, we require an unsupervised method also to determine the number of clusters. One solution that is commonly used in the literature is to find the number of clusters that optimize a clustering validation metric. There are many different formulations of this validation metric, but commonly two criteria considered; (1) the distance of the objects within the clusters should be minimized and (2) the distance between the clusters should be maximized. We used various formulations, for both data point clustering and model clustering steps which will be explained in next sections.

Data Point Clustering: [70] proposed a method that calculates 30 different clustering validation metric formulas and applies majority rule between them to determine the optimal number of clusters. Most of the formulas require the exact positions of the data points in the space, while some of them can be calculated using only the pairwise distance values between them. In data point clustering step, since we have the exact positions, we can calculate all 30 formulas which are explained in [70].

VOMM Clustering: In VOMM Clustering step, we do not have the spatial position information of the VOMMs in the feature space, instead, we only have the pairwise distances between VOMMs and therefore we used the metrics that only require the distances such as Silhouette Index, Frey Index, Dunn Index. For the sake of simplicity, we do not apply majority rule here and use the Frey Index [71] that gave the best empirical results.

Results: By using majority rule method for data point clustering and silhouette method for model clustering, up to 0.44 and 0.21 ARI is achieved on the Lego Demonstrator and Arçelik Press Machine data respectively.

Appendix B. Discretization on a Subset of the Data set and Interpolation on the Rest

To learn the data point clustering structure in VOMM MTSC, preprocessing (see 4.1) and discretization (see 4.2.1) steps can be applied on a subset of the data set instead of the complete data set. After the clustering is constructed, a classifier can be trained using the

clusters as classes and cluster predictions as class labels. Using the learned preprocessing parameters and the classifier, remaining data points in the data set can be classified. Naturally this approach assumes the subset of the data points can be a good representative sample of the underlying clustering structure of the data points of the modeled system. Our experiments shown that for both data sets, exactly same clustering structures and same MTSC results are obtained by learning such a classifier with only 25% of the data. The classifier that is learned was Nearest Neighbor Classifier [6]. This can be a promising remedy to high complexity of the discretization part in the VOMM MTSC complexity, uN^2 (see 5), since it is reduced h^2 times using a h times smaller data set. In this case of course, complexity of classification should be added to the overall complexity.

Appendix C. Applications on a Non-CPS Data Set

To test the applicability of the VOMM MTS clustering on MTS data from systems other than CPS, we applied the three presented MTS Clustering methods Human Activity Sensing Consortium (HASC) [72] data set. HASC provides data sets that are collected with the sensors that are placed on different agents while they perform different activities such as walking, jogging, skipping etc. Consortium organizes competitions where different methods proposed by the competitors try to identify the different types of movements by analyzing the sensory data.

To carry out experiments, we took a subset of the data set of the HASC 2014 competition [73] which contains 3 types of movements; walking, jogging and skipping, collected from 6 different people. 18 MTSs of have 3 classes since the goal of the competition is to identify different types of movements. There were three sensors placed on the agents therefore the data are three dimensional. The ranges of the parameters that are tested are same as the ones used on CPS data sets except that the dimensionality reduction is skipped since the dimensionality of the data set is low. The results of the experiments are shown in the Tables 9, 10 and 11.

We can observe from the tables that VOMM MTS clustering was again bound to mistakes, but it outperformed the other methods and gave promising results. It can also be seen from

the tables 7 and 9 that in the parameter settings that lead to success, there is a high similarity with the settings used for the experiments on the Arcelik data set. This is a sign that the range of the good values for the parameter are stable for various data sets or the data obtained from different sources. Best results are obtained when mk is set to 2 and analysis of the clustering structure reveals the reason; the MTSs correspond to jogging and walking fall into same cluster, but the skipping move is clearly distinguished from those two. From table 11 we see that HMM MTS clustering performs better than a random clustering for only a very specific combination of the parameter values. It is also observed from table 10 that PCA MTS clustering performed better than a random clustering for HASC data set for a high range of parameter settings, however, the level of the success is still limited.

Table 9: Adjusted Rand Index Results of VOMM MTS Clustering done with the combinations of parameters explained in table 2 on HASC Data. Adjusted RAND Index (ARI) is used for scoring.

t	L	mk	a=		4						5						6					
			k=	l=	2		3		4		2		3		4		2		3		4	
					0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2	0.1	0.2
3	5	2	-0.01	-0.01	-0.01	-0.05	0.01	0.01	0.35	-0.01	0.29	-0.03	0.04	0.13	0.2	0.04	0.25	0.05	0.25	0.35		
		3	0.12	0.12	0.12	0.14	-0.02	-0.02	0.12	0.12	0.14	0.09	0.08	0.18	0.11	0.02	0.26	0.01	0.18	0.26		
		4	0.03	0.03	0.05	0.11	-0.03	0.11	0.16	0.12	0.11	0.09	0.1	0.07	0.14	0.01	0.25	0.01	0.16	0.27		
	6	2	-0.01	-0.01	0.08	0.2	-0.07	0.01	0.35	0.04	0.29	0.15	0.04	0.25	0.2	0.04	0.35	-0.02	0.25	0.29		
		3	0.12	0.12	0.12	0.14	0.01	-0.02	0.26	0.12	0.2	0.14	0.06	0.24	0.13	0.02	0.2	0.03	0.18	0.29		
		4	0.06	0.06	0.03	0.11	0	0.27	0.16	0.16	0.17	0.1	0.07	0.03	0.08	0.17	0.2	0.03	0.25	0.3		
	7	2	0.35	-0.01	0.08	0.35	-0.07	-0.07	0.35	0.04	0.29	0.25	0.04	0.15	0.2	0.04	0.35	0.15	0.2	0.15		
		3	0.18	0.12	0.12	0.14	-0.02	0.2	0.12	0.12	0.14	0.14	0	0.24	0.13	0.05	0.2	0.01	-0.07	0.35		
		4	0.2	0.15	0.11	0.18	0	0.27	0.16	0.2	0.17	0.11	0.14	0.3	0.08	0.17	0.2	0.01	0.03	0.03		
4	5	2	-0.07	-0.01	-0.07	0.01	-0.07	-0.07	0.35	-0.01	0.15	0.15	0.04	0.04	0.35	0.08	0.25	0.15	0.54	0.05		
		3	0.12	0.12	0.02	0.12	-0.06	0.12	0.18	0.12	0.15	0.12	0.09	0.21	0.18	0.06	0.26	0.11	0.26	0.12		
		4	0.06	0.06	0.05	0.11	-0.07	0.11	0.11	0.12	0.11	0.09	0.11	0.17	0.2	0.04	0.2	0.04	0.29	0.21		
	6	2	-0.05	-0.01	0.01	-0.07	-0.07	-0.05	0.35	-0.01	0.15	0.15	0.15	0.04	0.35	0.08	0.54	0.05	0.46	0.05		
		3	0.12	0.12	0.02	0.12	-0.09	0.12	0.25	0.12	0.09	0.12	0.14	0.14	0.26	0.06	0.36	0.06	0.26	0.23		
		4	0.11	0.06	0.03	0.11	-0.07	0.11	0.16	0.25	0.1	0.12	0.07	0.2	0.2	0.17	0.2	0.01	0.27	0.21		
	7	2	0.35	-0.01	-0.07	-0.07	-0.07	0.35	0.35	-0.01	0.15	0.15	0.15	0.04	0.35	0.04	0.54	0.05	0.46	-0.04		
		3	0.29	0.12	0.02	0.12	-0.08	0.26	0.26	0.12	0.08	0.14	0.14	0.14	0.35	0.05	0.36	0.06	0.3	0.21		
		4	0.24	0.25	0.07	0.11	-0.1	0.35	0.16	0.2	0.08	0.12	0.14	0.2	0.2	0.11	0.28	0.01	0.17	0.21		
5	5	2	-0.07	-0.01	0.54	0.2	0.36	0.01	0.35	-0.01	0.13	0.13	0.13	-0.01	-0.02	-0.01	0.25	0.08	0.36	0.05		
		3	0.12	0.12	0.18	0.12	0.29	0.12	0.18	0.12	0.15	0.14	0.17	0.21	0.11	0.02	0.37	0.05	0.26	0.08		
		4	0.06	0.11	0.07	0.11	0.24	0.11	0.11	0.14	0.11	0.12	0.07	0.2	0.08	0.07	0.24	0.04	0.3	0.21		
	6	2	-0.05	-0.01	0.46	0.2	0.36	0.01	0.35	-0.02	0.13	0.13	0.25	-0.01	0.35	-0.01	0.36	0.05	0.36	0.01		
		3	0.12	0.12	0.26	0.12	0.2	0.26	0.2	0.14	0.15	0.12	0.17	0.09	0.26	0.02	0.35	0.06	0.12	0.21		
		4	0.16	0.16	0.11	0.11	0.17	0.35	0.11	0.14	0.1	0.12	0.07	0.13	0.2	0.17	0.37	0.07	-0.01	0.21		
	7	2	0.2	-0.01	0.2	0.2	0.29	0.29	0.35	0.15	0.13	0.15	0.25	-0.01	0.35	-0.02	0.08	0.06	-0.05	0.01		
		3	0.18	0.12	0.37	0.12	0.27	0.2	0.26	0.14	0.11	0.14	0.17	0.09	0.13	0.03	0.09	0.09	0.21	0.21		
		4	0.2	0.16	0.33	0.11	0.24	0.28	0.16	0.15	0.08	0.12	0.14	0.2	0.2	0.15	0.21	0.02	0.14	0.11		

Table 10: Adjusted Rand Index Results of PCA MTS Clustering done with the combinations of parameters explained in table 2 on HASC Data. Adjusted RAND Index (ARI) is used for scoring.

DS	D	$\alpha=$ mk=	0.4						0.5						0.6						
			3	4	6	7	13	14	3	4	6	7	13	14	3	4	5	6	7	13	14
8	2		0.01	0.03	-0.04	-0.03	0.03	0.00	0.01	0.03	-0.04	-0.03	0.03	0.00	0.01	0.03	0.00	-0.04	-0.03	0.03	0.00
	3		0.10	0.15	0.08	0.08	0.10	0.04	0.10	0.15	0.08	0.08	0.10	0.04	0.10	0.15	0.15	0.08	0.08	0.10	0.04
9	2		0.01	0.03	0.05	0.02	0.03	0.00	0.01	0.03	0.05	0.02	0.03	0.00	0.01	0.03	0.00	0.05	0.02	0.03	0.00
	3		0.10	0.15	0.08	0.08	0.22	0.12	0.10	0.15	0.08	0.08	0.22	0.12	0.10	0.15	0.15	0.08	0.08	0.22	0.12
10	2		-0.10	-0.01	-0.01	-0.02	0.03	0.00	-0.10	-0.01	-0.01	-0.02	0.03	0.00	-0.10	-0.01	-0.02	-0.01	-0.02	0.03	0.00
	3		0.10	0.15	0.11	0.08	0.22	0.12	0.10	0.15	0.11	0.08	0.22	0.12	0.10	0.15	0.15	0.11	0.08	0.22	0.12
11	2		0.03	0.06	0.01	0.04	0.03	0.00	0.03	0.06	0.01	0.04	0.03	0.00	0.03	0.06	0.04	0.01	0.04	0.03	0.00
	3		0.10	0.15	0.11	0.08	0.22	0.12	0.10	0.15	0.11	0.08	0.22	0.12	0.10	0.15	0.15	0.11	0.08	0.22	0.12
12	2		0.23	0.01	-0.01	0.07	0.03	0.00	0.23	0.01	-0.01	0.07	0.03	0.00	0.23	0.01	-0.02	-0.01	0.07	0.03	0.00
	3		0.10	0.15	0.23	0.08	0.10	0.04	0.10	0.15	0.23	0.08	0.10	0.04	0.10	0.15	0.15	0.23	0.08	0.10	0.04
13	2		-0.02	0.03	0.01	0.04	0.03	0.00	-0.02	0.03	0.01	0.04	0.03	0.00	-0.02	0.03	0.01	0.01	0.04	0.03	0.00
	3		0.10	0.15	0.23	0.08	0.22	0.12	0.10	0.15	0.23	0.08	0.22	0.12	0.10	0.15	0.15	0.23	0.08	0.22	0.12
14	2		-0.02	0.03	0.00	0.04	0.03	0.00	-0.02	0.03	0.00	0.04	0.03	0.00	-0.02	0.03	0.04	0.00	0.04	0.03	0.00
	3		0.10	0.15	0.08	0.08	0.06	0.00	0.10	0.15	0.08	0.08	0.06	0.00	0.10	0.15	0.15	0.08	0.08	0.06	0.00
15	2		0.03	0.03	0.19	0.04	0.03	0.04	0.03	0.03	0.19	0.04	0.03	0.04	0.03	0.03	0.12	0.19	0.04	0.03	0.04
	3		0.10	0.15	0.23	0.08	0.06	0.12	0.10	0.15	0.23	0.08	0.06	0.12	0.10	0.15	0.15	0.23	0.08	0.06	0.12

Table 11: Adjusted Rand Index Results of HMM MTS Clustering done with the combinations of parameters explained in table 2 on HASC Data. Adjusted RAND Index (ARI) is used for scoring.

S	W	$a=$ mk=	4			5			6			7		
			3	4	5	3	4	5	3	4	5	3	4	5
2	3		-0.10	-0.10	-0.10	-0.02	0.12	0.07	0.00	-0.01	-0.01	-0.01	-0.02	-0.01
	4		-0.10	0.11	0.12	-0.02	0.12	0.07	0.20	-0.01	-0.01	-0.01	-0.02	-0.01
	5		0.29	0.11	0.12	-0.02	-0.03	0.08	0.20	0.23	0.23	-0.01	-0.02	-0.01
	6		0.29	0.11	0.12	-0.02	-0.03	0.08	0.20	0.23	0.23	-0.01	-0.02	-0.02
	7		0.29	0.11	0.12	-0.02	-0.03	0.07	0.00	0.16	0.19	-0.01	-0.02	-0.02
3	3		0.05	-0.01	-0.03	-0.02	-0.07	-0.07	-0.03	-0.10	-0.09	0.02	-0.01	-0.03
	4		0.01	-0.01	-0.03	-0.02	-0.07	-0.07	-0.05	-0.10	-0.09	0.02	-0.01	-0.03
	5		-0.01	-0.01	-0.03	-0.02	-0.07	-0.07	-0.05	-0.10	-0.09	-0.01	-0.03	-0.03
	6		-0.01	-0.01	-0.10	-0.02	-0.03	-0.04	-0.05	-0.10	-0.09	-0.01	-0.03	-0.03
	7		-0.01	-0.01	-0.10	-0.02	-0.03	-0.04	-0.05	-0.10	-0.09	-0.01	-0.03	-0.03
4	3		-0.03	0.11	-0.03	-0.01	-0.05	-0.03	-0.03	-0.02	-0.02	-0.03	-0.05	-0.09
	4		0.02	-0.01	-0.01	-0.01	-0.05	-0.03	-0.03	-0.03	-0.02	-0.03	-0.05	-0.09
	5		0.02	-0.01	-0.01	-0.01	-0.05	-0.03	-0.03	-0.03	0.00	-0.01	-0.05	-0.09
	6		0.02	-0.01	0.10	-0.01	-0.05	-0.03	-0.03	-0.03	0.00	-0.01	-0.05	-0.09
	7		0.02	-0.01	0.10	-0.01	-0.05	-0.03	-0.03	-0.03	-0.02	-0.01	-0.05	-0.09
5	3		-0.04	-0.02	-0.01	-0.06	-0.02	-0.04	-0.03	-0.04	-0.07	-0.02	-0.05	-0.05
	4		-0.04	-0.02	-0.01	-0.06	-0.02	-0.05	-0.03	-0.04	-0.07	-0.02	-0.05	-0.05
	5		-0.04	-0.02	-0.01	-0.06	-0.02	-0.05	-0.02	-0.05	0.05	-0.02	-0.05	-0.05
	6		-0.04	-0.02	-0.01	-0.06	-0.02	-0.05	-0.02	-0.05	0.05	-0.02	-0.05	-0.05
	7		-0.04	-0.02	-0.01	-0.06	-0.02	-0.05	-0.02	-0.05	0.12	-0.02	-0.05	-0.10
6	3		-0.03	-0.02	-0.04	0.01	-0.02	0.02	-0.05	-0.05	0.04	-0.05	-0.05	-0.09
	4		-0.03	-0.02	-0.04	0.01	-0.02	-0.01	0.12	0.10	0.09	-0.05	-0.04	-0.08
	5		-0.03	-0.02	-0.04	-0.02	-0.05	-0.04	-0.01	0.01	-0.02	-0.05	-0.04	-0.08
	6		-0.03	-0.02	-0.04	-0.02	-0.05	-0.04	-0.01	-0.05	-0.02	-0.05	-0.04	-0.08
	7		-0.03	-0.04	-0.08	-0.02	-0.05	-0.04	-0.01	-0.05	-0.02	-0.03	-0.04	-0.08

We conclude that, VOMM MTS clustering method is a promising candidate to be applied on data from various sources and this is confirmed with the results of the experiments on a real-world data set of human activity sensing field.

Bariş Gün Sürmeli

Address: inIT – Institute Industrial IT
Technische Hochschule Ostwestfalen-Lippe
Campusallee 6 32657 Lemgo
Phone: + 49 (0) 5261 / 702 – 5754
Email: baris.suermeli@th-owl.de

SUMMARY

An artificial intelligence enthusiast who likes to develop machine learning methods and apply them in the industry.

WORK EXPERIENCE

July 2018 - Present

Research Assistant, SMARTPas Research Project, Institute Industrial IT, Lemgo, Germany

- Working on developing and applying Machine Learning methods on data obtained from food and beverages with the aim of safer sterilization.

Jan 2016 - July 2018

Researcher, IMPROVE Research Project, Marmara University, Turkey

- Worked on developing and applying Machine Learning methods with the aim of cost reduction in industrial Cyber-Physical Systems.

June 2015 – July 2015

Trainee, Marmara University, Turkey

- Worked on manual co-referencing In Turkish Natural Language with Prof. Dr. Peter Schüller. Improved the capabilities of an existing NLP tool (GATE). Annotated co-references in articles and co-authored a Turkish Co-reference Annotation Manual.

EDUCATION

2016–Present
MSc. Computer Science, Marmara University, Istanbul, Turkey

2012 - 2015
BSc. Computer Science & Engineering, Marmara University,
Istanbul, Turkey

PUBLICATIONS

2017
Sürmeli, B. G., Eksen, F., Dinç, B., Schüller, P., & Tümer, B. (2017, July). Un-supervised mode detection in cyber-physical systems using variable order Markov models. In Industrial Informatics (INDIN), 2017 IEEE 15th International Conference on (pp. 841-846). IEEE.

2017
Schüller, P., Cıngıllı, K., Tunçer, F., Sürmeli, B. G., Pekel, A., Karatay, A. H., & Karakaş, H. E. (2017). Marmara Turkish Coreference Corpus and Coreference Resolution Baseline. arXiv preprint arXiv:1706.01863. (under minor revision at Natural Language Engineering)