COMPUTATIONAL ALGORITHMS

FOR PREDICTIVE HEALTH ASSESSMENT

A Dissertation

presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

Department of Computer Science
College of Engineering

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

ZAHRA HAJIHASHEMI

Dr. Mihail Popescu, Dissertation Supervisor

May 2015

The undersigned, appointed by the dean of the Graduate School, have examined the dissertation entitled

Computational Algorithms for Predictive Health Assessment

presented by Zahra Hajihashemi,

a candidate for the degree of doctor of philosophy, and hereby certify that, in their opinion, it is worthy of acceptance.

Associate Professor Mihail Popescu

Associate Professor Toni Kazic

Associate Professor Jianlin Jack Cheng

Professor Marilyn Rantz

Dedication

This thesis work is dedicated to my husband, Mohammad, who has been a constant source of support and encouragement during the challenges of graduate school and life. I am truly thankful for having you in my life. This work is also dedicated to my parents, Jalil and Shahnaz, who have always loved me unconditionally and whose good examples have taught me to work hard for the things that I aspire to achieve.

Acknowledgements

I would like to express the deepest appreciation to my dissertation supervisor, and the committee chair Dr. Mihail Popescu, who has the attitude and the substance of a genius: he continually and convincingly conveyed a spirit of adventure in regard to research and scholarship, and an excitement in regard to teaching. Without his guidance and persistent help this dissertation would not have been possible.

I would like to thank my committee member, Professor Rantz, who put her hard works and leadership to establish TigerPlace and provided such an incredible research opportunity for students. I would like to thank my committee member, Dr. Kazic who thought me the concepts of Bioinformatics and computational biology. I would like to thank my committee member, Dr. Cheng who thought me machine learning and data mining.

In addition, a special thanks to my parents Jalil and Shahnaz, who went behind their way to raise me and send me to the US and paid for my education. Worm thanks to my siblings Nahid, Zohreh, Hamid, and Mohsen who loved me unconditionally and supported me in every way they could. I would like to thank my nieces Minoo, and Sara, and my nephews Ehsan, Amin, Ali, and Reza who supported me emotionally and brought smile to my face during the ups and downs of my way. I would also like to thank the family of my husband, Hossein, Zahra, Ali, and Mina for their love and their support.

Last but not least, I would like to thank my husband, Mohammad, who was with me, supported me, and loved me in the every step of my way. He helped me to overcome challenges and problems during my dissertation. I'm very thankful for having him in my life.

Contents

List of Tables

List of Figures

List of Abbreviations

AAN	American Academy of Nursing
ADL	Activity of Daily Living
AIP	Aging in Place
ANOVA	ANalysis Of VAriance
BFO	Bacterial Foraging Optimization
CERT	Center for Eldercare and Rehabilitation Technology
CUI	Concept Unique Identifier
DTW	Dynamic Time Wrapping
EHR	Electronic Health Record
EM	Expectation Maximization
EIR	Early Illness Recognition
FAR	False Alarm Ratio
FCM	Fuzzy C-Means
FIs	Frequent Item sets
FLAME	Flexible and Accurate Motif Detector
GMM	Gaussian Mixture Model
HMM	Hidden Markov Models
ICD-9	Classification of Disease version 9
LCSS	Longest Common Subsequence
LR	Logistic Regression
LUI	Lexicon Unique Identifier
MAP	Maximum A Posterior probability
MATS	Multi Attribute Sensor Time Series
MD-DTW	Multi-Dimensional Dynamic Time Warping
MLP	Multilayer Perceptron
NLP	Natural Language Processing
POD	Probability Of Detection
RF	Random Forest
RMS	Root Mean Square

SMOTE	Synthetic Minority Over-sampling TEchnique
SW	Smith Waterman
TSW	Temporal Smith-Waterman
UMLS	Unified Medical Language System

COMPUTATIONAL ALGORITHMS

FOR PREDICTIVE HEALTH ASSESSMENTS

Zahra Hajihashemi

Dr. Mihail Popescu, Dissertation Supervisor

Abstract

Rapid aging of the population in the US requires increased attention from health care providers and from the entire society as a whole. While the elderly population (aged over 65) will increase by 8% until 2050 in the developed countries, the working-age population (age between 15 and 64 years) will decrease and its ratio to the elderly population will decline from 4.3 to 2.3 [1]. While older adults prefer to live independently, many of the health conditions associated with old age, such as frailty, dementia, and risk of falling require increased attention and monitoring. However, independent living may lead to infrequent health assessments due to lack of continuous monitoring or fear of being institutionalized. Late health assessments may miss unreported complications, which in turn lead to poor long-term prognosis and quality of life [2]. A possible solution to prevent unreported health problems in independently living older adults is through automatic health monitoring systems.

The aim of this dissertation is to use sensor network technology to detect changes in health status of elderly living alone, alert health care providers, and augment traditional health care. We review the Aging In Place (AIP) research and the sensor technology developed at the University of Missouri (MU) to support AIP. AIP represents one's ability to live in his/her own home safely and independently regardless of age, income, or medical condition.

In this dissertation, we address two topics. First, we discuss the problem of measuring the temporal similarity of two multi-dimensional time series. The second topic of this work is predicting health patterns using time series similarities. For measuring the similarity of multi-dimensional time series, we focus on health care applications. We review already discovered similarity functions for time series. Inspired by the bioinformatics approach, we propose a new method to measure the similarity of two multi-dimensional time series. We introduce the idea of

modified Smith-Waterman framework, Temporal Smith-Waterman (TSW), for Early Illness Recognition (EIR) that uses temporal sequences. We analyze the properties of TSW, introduce a faster way to compute it based on genetic programming, and propose a new method for searching large time series. Our approach doesn't require series data conversion to continuous format. Instead, we arrange all different sensor hits together with their time stamps into a one-dimensional sequence. Our method overcomes difficulties related to data uncertainty and aggregation that often arise when processing sensor data.

For health pattern recognition, we describe a new framework that uses TSW for abnormal pattern recognition and its application to eldercare and early illness recognition. The new framework for predicting abnormal health patterns uses non-wearable sensors and sensor pattern similarity. A sensor pattern is classified as "abnormal" if it is much less similar to the previous ones. In this work, we consider "abnormal" days as days with documented nurse assessment in the electronic health record. To further refine the nature of the health assessment used in our EIR framework, we proposed a novel semi-supervised for clustering the nursing notes. This method represents each cluster of notes by a language model. We use the bi-grams model of terms to represent term to term dependencies and improve the expressivity of the proposed clustering method. We estimate the cluster model by a maximum a posterior probability (MAP) to address the challenge of highly imbalance nursing visit notes dataset.

In this work, we also propose three methods for identification of deviations in patterns of activities of daily livings (ADL) of older adults and use them to generate alerts for the healthcare providers. ADLs such as bathroom visits can be monitored by automated in-home sensor systems. Our proposed methods find periodicity in sensor time series data using clustering, item set mining, and statistical approaches. A retrospective multiple case study (N=3) design was used to quantify bathroom visits as parts of the older adult's daily routine, over a 10-day period. The performances of the proposed methods are evaluated using data collected from TigerPlace, an independent living community situated in Columbia, Missouri.

Chapter 1 Introduction

The American Academy of Nursing (AAN), in 1996, asked researchers to propose new solutions to modify and enhance the standards of eldercare services [3]. In many developed countries the rapid aging of the population garnered attention from health care providers. Published statistics show an increase in the elderly population (aged 65 and older) from 13% of the population in 2010 to 19% in 2030, while the working-age population (age between 15 and 64 years) will decrease and its ratio to the elderly population will decline from 4.3 to 2.3 [4]. Older adults are eager to live independently, regardless of complex chronic conditions such as frailty, dementia and risk of falling. However, independent living may in turn lead to delayed health assessments due to the lack of monitoring, which is associated with poor long-term health outcomes [2]. Late health assessment is an aggravating risk factor that usually occurs because of the fear of institutionalization and the failure of a physician's assessments [5]. A possible solution to unreported health problems is to utilize an automatic health monitoring system that is able to detect and report signs of early illness.

Sensor networks are a promising approach to health monitoring. In the last decades, a growing number of projects have deployed ubiquitous sensor networks to monitor health of older adults. MIT PlaceLab, Georgia Tech's aware House, Honeywell's Independent Lifestyle Assistant, and The University of Missouri's TigerPlace (a senior housing complex that enables residents to age in place) are such successful examples [6] [7] [8] [9]. A variety of methodologies for detecting activity and assess medication compliance have been reported in the literature [10] [11] [12] [13]. A key feature of these health-monitoring systems is the ability to continuously and unobtrusively collect information about daily activities of older adults. These systems process the acquired sensor data, infer the activities and behaviors of older individuals, and detect changes in their health status. Considering the health context of the monitored patient is still an unsolved problem.

Health context of each individual elderly is a very important factor to be considered, since normal behaviors can change due to hospitalizations, whereas sleep

patterns or fall risks can be affected by changes in medications. In this dissertation, we develop health context aware monitoring systems that have the power to detect and predict early signs of illness and functional decline using embedded sensor data.

1.1. The Problem

Health information technology has been used in long-term care to improve outcomes and reduce costs. In TigerPlace, an aging in place facility from Columbia, MO, we are interested in detecting early signs of illness using sensor networks and an Electronic Health Record (EHR) system. The aim of this work is to investigate early illness prediction algorithms based on non-wearable sensor data and medical concepts extracted from nursing notes employing Natural Language Processing (NLP) methodologies. There are several challenges in this research that we intend to address.

- We need to define a suitable distance function to measure the dissimilarity of multi-attribute time sequences.
- We need to find association algorithms between the sets of Unified Medical Language System (UMLS) concepts and daily sensor sequences.
- We should employ aggregation methods to infer the most probable health concepts for an unknown sensor sequence.
- We should predict the change in resident's health status based on sensor data and EHR data.
- We need to employ a motif discovery approach to find episodes in sensor data that can be applied into sensor frequent pattern mining. Finding frequent patterns (motifs) in sensor sequences leads to predict unseen patterns in sensor time series.

Time series or sequence data sets arise in many real world applications like stock market, weather forecast, video databases, sensor-based controls, and medicine. In healthcare domain, medical data is usually multi-attribute, since a single sensor is rarely sufficient to record the health of a patient: such a sequence can be obtained by using the blood pressure values, the heart beat rate, and other parameters recorded periodically from a patient. In smart homes equipped with a sensor network, the data collected from

multiple sensors in different days results in a multi-attribute, temporal dataset. Predicting similar health patterns from such a dataset is still an open question. Following, we outline the contributions of our work to address the challenges we mentioned before.

1.2. Contributions

As mentioned earlier, health pattern detection is an important task which has been widely used in many health care and health prevention applications. Most of the discussion in this document will revolve around three key tasks to predict health patterns based on sensor data including defining a suitable similarity measure for two multi attribute sensor time series data, recognizing abnormal health events based on daily sensor time series, and annotating sensor time series that involves inferring the most probable health concepts for an unknown sensor sequence. Following, we briefly discuss the contributions of this dissertation.

1.2.1. Temporal Smith Waterman algorithm

The task of assessing similarities of behaviors can be cast as computing the similarity of Multi Attribute Time Series (MATS). MATS similarity is a challenging task and is still an open research question. To address this challenge, we propose a novel approach that considers MATS as one-dimensional sequence and to use bioinformatics techniques to find the best alignment. This approach is possible when sensor data is already quantized by the hardware system which results in a symbolic sequence of sensor firings and the related firing time. Finding local similarity in sensor time series is desirable, since it shows similar health patterns in the local sequence regions. However, for health sensor data, the time stamp is an important factor that needs to be considered when finding optimal local time series alignments. Smith Waterman (SW) algorithm is a well-known algorithm to find an optimal local alignment [14]. However, it does not consider time in alignment. In this research, we propose a new version of the SW algorithm, Temporal Smith-Waterman, which considers time in the scoring schema. This approach allows finding the similarity of two MATS and their optimal alignment with respect to time, an important factor in interpreting sensor data. We compare our method

with an approach that uses sensor aggregation and an Euclidean-type (Root Mean Square Error- RMSE) similarity measure.

1.2.2. Analysis on Temporal Smith-Waterman algorithm

We demonstrate the properties of TSW for early illness recognition that uses time-stamped sequences. Moreover, we review the bottlenecks of TSW and discuss ways to improve its performance in health pattern detection. In the original TSW method, we represented the sensor time series of each 24-hour day as a one-dimensional sequence and compared sequences for two different days. However, the question is whether this generalizes to comparing sensor sequences that are only several hours long (e.g. 5 hours)? To solve this problem, we introduce a faster way to compute the similarity of two MATS based on genetic programming and propose a new method for searching large time series. In the end, we show how TSW similarity can be used for detecting abnormal health pattern using non-wearable sensor networks.

We then apply our method to sensor data from TigerPlace for early illness detection. Our method overcomes difficulties related to data uncertainty and aggregation that often arise when processing sensor data. The experiments take place at aging-in-place facility, TigerPlace in Columbia, MO. We provide a set of experiments that investigate TSW properties on artificial datasets together with experiments on TigerPlace datasets. On a pilot sensor dataset from nine TigerPlace residents, with a total of 1902 days of collected data, our proposed method achieves an average F-measure of 80%. Moreover, we validate our proposed framework on a well-known sensor time series bench mark dataset, localization data for posture recognition [15]. On this bench mark dataset, our method obtains about 82% F-measure on posture recognition, which shows an improvement of 10% compare to the results published in [15].

1.2.3. Health pattern detection using TSW

One of the aims of this dissertation is to predict the change in resident's health status based on sensor data produced by in-home monitoring system. We represent a novel approach to use TSW as similarity function to classify sensor patterns as "normal" or "abnormal" days. "Abnormal" days are defined by unusual sensor activity patterns that

require a nurse's assessment of the resident where there are nursing documentations associated with these days in the HER system. We explore the detection of abnormal patterns by employing the distribution of similar sensor sequences. We assume that an abnormal sensor pattern is due to a medical condition, fact that is true in general for the elderly population from TigerPlace. The difficulty of finding similar sensor patterns is given by the multidimensional nature of sensor data and their huge volume. We use the similarity measurement, TSW, and introduce a new confidence level for the purpose of *k*-nearest neighbor classification. We compare our results with binary *k*-nearest neighbor. On the TigerPlace pilot data set, the proposed classification approach has an average precision of 0.70 and a recall of 0.30. In this comparison, the proposed TSW framework outperforms binary *k*-nearest neighbor approach.

1.2.4. Sensor sequence annotation using TSW and NLP

The healthcare prevention saves billions of dollars each year for healthcare institutes. To improve the efficiency of early illness recognition system, it is crucial to not only detect but also annotate an abnormal event with the most possible health issues. This helps healthcare providers to plan ahead for an upcoming abnormal event. However, it requires an expensive process and hours of the medical expert's times and manual works. We propose a new semi-supervised learning framework for automated early illness recognition system.

We design a novel framework that detects a change in resident's health status and predicts the possible health issues based on sensor data and nursing notes produced by an integrated in-home monitoring system. We propose a novel semi-supervised clustering method that automatically uses user provided information to effectively cluster nursing notes. This framework represents each note cluster by a language model. We use the bi-gram model of terms to improve the expressivity of the proposed clustering method. We estimate the cluster model by a MAP.

Our system achieves high accuracy on medical nursing notes, 0.75, while other baseline methods such as Expectation Maximization (EM), hierarchical clustering, and *k*-means have accuracy of 0.62 in average. A set of experiments on a benchmark dataset,

20-News Group, results the high accuracy of 0.8. We highlight how to address the challenge of clustering on an imbalance, overlapping medical document dataset using the proposed semi-supervised learning framework.

1.2.5. Detecting Daily Routines Using a Clustering Approach

For the aim of EIR, the living environment can be equipped with passive sensor networks to monitor the daily life of the older adult. Functional ability can be reflected in the performance of routine tasks that include activities of daily living (such as bathing, dressing, hygiene, and bowel movement), more complex instrumental activities of daily living (s. a. housework, finance management, and shopping), and general life activities (s. a. hobbies, leisure past time and socialization). Routines develop within temporal patterns of day and night, weekday and weekend, and seasonal change. This periodicity of routine behaviors can be modeled with mathematical algorithms. Extracting and analyzing attributes of daily routines may be useful in understanding trajectories of functional decline of older adults and can suggest timely interventions.

Our goal is to develop an algorithm to identify deviations in patterns of day-to-day activities of older adults to generate alerts to the healthcare providers for timely interventions. Daily routines, such as bathroom visits, can be monitored by automated in-home sensor systems. We present a novel approach that finds periodicity in sensor time series data using a clustering approach. A retrospective multiple case study (N=3) design was used to quantify bathroom visits as parts of the older adult's daily routine, over a 10-days period. The distribution of duration, number, and average time between sensor hits was used to define the confidence level for routine visit extraction. Then, a hierarchical clustering was applied to extract periodic patterns. The performance of the proposed method was evaluated through experimental results.

1.2.6. Frequent Temporal Mining on Time Series

Typical EIR approaches are either concentrated on the detection of a given set of activities such as a fall or walks, or on the detection of anomalies such as too many bathroom visits. The decline of functional ability of older adults in activities such as ability to shower, ability to dress, etc can be captured by in-home monitoring systems.

Some functional declines are temporary such as the ability to shower after having a leg injury, before they become permanent. Our goal with sensor-based health assessment is to stop the decline through early recognition and/or prediction of health problems in advance. We propose a new illness recognition framework based on the detection of a missing frequent activity (MFA) (as opposed to detecting an abnormal activity).

Under this framework, at the end of the day, we examine if all frequent patterns found in previous two weeks are still present. If some patterns are found missing several days in a rows, we assume that it is due to an early onset of a decline, and we send an alarm to caregiver. To detect the loss of an existent ability, we need to employ a frequent pattern detection method that is independent of the types and number of activities performed by the resident. A novel MFA is proposed using a frequent temporal pattern detection algorithm and demonstrated on a pilot dataset collected in TigerPlace.

1.3. Outline

This chapter gives an overview of this dissertation by stating the problem that is focused around health pattern detection, current and up-to-date work that I have contributed to the research community and an outline that describes the overall organization and content of this document.

Chapter 2 represents the system architecture and the dataset that we used for this research. In the first part of this chapter, we briefly describe the sensor network architecture and type of sensors that we used. We review different approaches to store and retrieve data. In addition, we present the preliminary studies and experimental results we have done on TigerPlace dataset. We discuss the limitations of the current approaches for time series similarities in health pattern detection.

Chapter 3 provides a review on literatures related to EIR systems. We discuss published articles in different time series data mining aspects such as supervised learning, unsupervised learning, indexing, similarity measure, and motif discovery.

Chapters 4 and 5 cover our novel idea of TSW, a temporal similarity measure for sensor time series. We describe the proposed method and illustrate a new framework that uses TSW as a metrics for health pattern detection. We present our experimental results

in detecting abnormal days. Moreover, for each sensor sequence, we associate health concepts extracted from the nursing notes using Metamap, a NLP tool provided by UMLS. In chapter 5, we address the challenges of finding the most suitable time sequence similarity and aggregation of the retrieved UMLS concepts.

In chapter 6, we use TSW to predict the change in resident's health status based on sensor data. We explore the detection of abnormal patterns by employing the distribution of similar sensor sequences. We use the similarity measurement, TSW, and introduce a new confidence level for the purpose of classification.

Chapter 7 describes a novel framework that detects a change in resident's health status and predicts the possible health issues based on sensor data and nursing notes produced by the integrated in-home monitoring system. We explain the proposed semi-supervised clustering method that automatically uses user provided information to effectively cluster nursing notes.

Chapter 8 shows a comparison on different methods in EIA system implemented at TigerPlace for health prediction purposes. We discuss the results of our approaches in EIR on TigerPlace dataset and compare them with the Gaussian approach which is implemented and running now at TigerPlace.

Chapter 9 covers the work in frequent activity pattern mining (motif discovery). We talk about sensor episode mining and its application in health pattern detection. We present three approaches on motif discovery.

We finish this dissertation in chapter 10. We conclude the results of our proposed methods and discuss their performances and limitations. We provide directions for future researches.

1.4. Publications

Journal Publications

1. Zahra Hajihashemi, Mihail Popescu, "*A Multidimensional Time Series Similarity Measure with Applications to Eldercare Monitoring*", **The IEEE Journal of Biomedical and Health Informatics**, 2014, Under Revision.
2. Zahra Hajihashemi, Mihail Popescu, "*A Semi-supervised Learning Framework for Unbalanced Medical Report Dataset with Language Modeling*", **The IEEE Transaction on Cybernetics**, 2015, Under Review.

Conference Publications

1. Zahra Hajihashemi, *M*ihail Popescu, "*A New Illness Framework Using Frequent Temporal Pattern Mining*", **ACM UbiComp International Workshop on Smart Health Systems and Applications**, September 13-17, 2014, Seattle, USA.
2. Zahra Hajihashemi, Mihail Popescu, "*Detecting Daily Routines of Older Adults Using Sensor Time Series Clustering*", **36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC'14)**, August 26-30, 2014, Chicago, USA.
3. Zahra Hajihashemi, Mihail Popescu, "*Detection of Abnormal Sensor Patterns in Eldercare*", **IEEE International Conference on E-Health and Bioengineering (EHB)**, November 21-23, Iasi, Romania, 2013.
4. Zahra Hajihashemi, Mihail Popescu, "*An Early Illness Recognition Framework Using a Temporal Smith Waterman Algorithm and NLP*", **AMIA Annual Symposium**, November 16-20, 2013, Washington, DC, USA.
5. Zahra Hajihashemi, Mihail Popescu, "*Predicting Health Patterns Using Sensor Sequence Similarity and NLP*", **IEEE International Conference on Bioinformatics and Biomedicine (BIBM)**, October 4-7, 2012, Philadelphia, USA.

Chapter 2 State of the Art

Rapid aging of the population in the US requires increased attention from health care providers and from the entire society as a whole. The elderly population (aged over 65) will increase by 8% until 2050 in the developed countries, whereas the working-age population (age between 15 and 64 years) will decrease and its ratio to the elderly population will decline from 4.3 to 2.3 [1]. In the same time, older adults prefer to live independently, many of the health conditions associated with old age, such as frailty, dementia, and risk of falling require increased attention and monitoring. However, independent living may lead to infrequent health assessments due to lack of continuous monitoring or fear of being institutionalized. Late health assessments may miss unreported complications, which in turn lead to poor long-term prognosis and quality of life [2]. A possible solution to prevent unreported health problems in independently living older adults is through automatic health monitoring systems.

The aim of this work is to use sensor network technology to detect changes in health status of elderly living alone, alert health care providers, and augment traditional health care. In this chapter, we review the AIP research and the sensor technology developed at MU to support AIP. AIP represents one's ability to live in his/her own home safely and independently regardless of age, income, or medical condition [16].

In TigerPlace, smart home technologies were developed to support older adults living alone and help them maintain independence [17]. The Center for Eldercare and Rehabilitation Technology (CERT) from the College of Engineering was established at MU to investigate, develop, and evaluate technology to help older adults and others with physical and cognitive challenges with their needs. A group of interdisciplinary faculty, students, and staff from nursing, computer engineering, computer science, health informatics, social work, physical therapy, and medicine are working together at CERT to test new technologies at TigerPlace with the help of the residents and their families. In the next section, we describe the system architecture of the monitoring system deployed at TigerPlace.

2.1. System Architecture

Technology has a tremendous impact on elderly by offering them full productive and independent lives. In TigerPlace, sensor technology has been utilized to help elderly residents not only manage their illness but also stay as healthy and independent as possible. An integrated monitoring system was deployed in 47 TigerPlace apartments with the University of Missouri IRB approval. TigerPlace residents are typical older adults. There are 64 residents ranging in age from 66 to 98 (mean 87.19). There are 11 couples and the remaining residents are single. Most residents have at least one chronic disease, including heart disease, diabetes, osteoarthritis, or early-stage dementia. The residents take an average of 12.5 medications; 8 residents use wheelchairs and 28 residents use walkers. Most residents ambulate with no assistance [18].

After focus groups with TigerPlace residents early on in our research [9], we decided to use only non-wearable sensors for monitoring, since they are unobtrusive, reliable, user friendly, able to detect a range of emergencies, and require no or minimal action on the part of the user, and importantly more acceptable by older adults. The monitoring has started in the fall of 2005. On average, there are two years of data for each resident. Figure 2-1 shows the architecture of our data acquisition system.

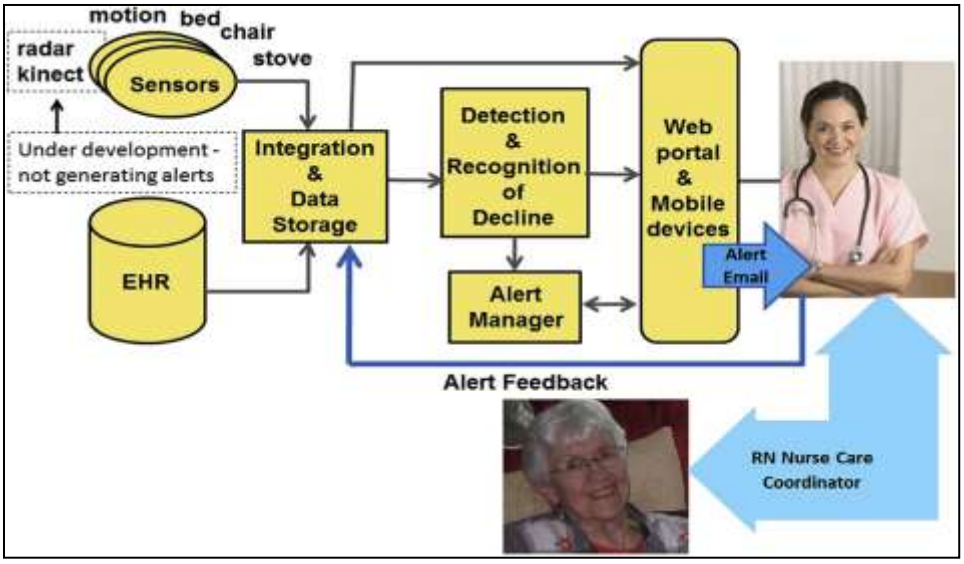


Figure 2-1. Integrated sensor network under development at TigerPlace.

The main components of the monitoring system are: an integrated motion/video/Kinect sensor network, a data logger, a reasoning system that analyses sensor patterns, an EHR system, an alert manager to notify clinicians of potential problems, and a secure Web-based interface to display the data for the clinicians and researchers.

The sensor network includes (1) stove temperature, bed, chair, and motion sensors; (2) pulse-Doppler radar and Microsoft Kinect sensors (Microsoft, Redmond, WA), which are being tested and developed. The activities of the residents in their apartment are monitored by the inexpensive passive infrared motion sensors. Using sensor motion data specific activities can be detected. For example, bathing activities are monitored by a sensor located above the shower, and food preparation activities are detected by sensors installed in kitchen cabinets and the refrigerator. The stove temperature sensor implies cooking activities and can be used to generate an alert if the stove has been left on too long by a forgetful resident. Sleep patterns are monitored by the bed sensors that measure qualitative pulse, respiration, and restlessness while the resident sleeps. The current bed sensor is a pneumatic strip which lies on top of the mattress under the sheets [19]. Table 2-1 lists our sensors.

Table 2-1. ID of the 23 sensor firings used in this research.

ID | Sensors firings	6. Breathing2	12. Living room Motion	18. Drawer
1. BedMovement1	7. Breathing3	13. Bathroom Motion	19. Cabinet
2. BedMovement2	8. Pulse1	14. Off Chair	20. Cup Cabinet
3. BedMovement3	9. Pulse2	15. On Chair	21. Refrigerator
4. BedMovement4	10. Pulse3	16. Stove Temp High	22. Plate Cabinet
5. Breathing1	11. BedroomMotion	17. Stove Temp Low	23. Silverware Drawer

A physical sensor can have multiple firing types, because a physical sensor can be complex (such as the bed sensor or the vision sensors) and have multiple firings. For example, bed sensors (firings with ID 1-10) can detect four levels of motion in bed "BedMovement1-4" (less than 3 seconds (s), 3 to 7s, 7 to 15s, greater than 15s), three levels of breathing "Breathing1-3" (low, normal, high), and three levels of pulse "Pulse1-

3” (low, normal, high). Motion sensors installed in bedroom, living room, bathroom, and chair (Firings with ID 11 to 15) are all produced by motion sensors, while firings with ID 16 and 17 are given by a temperature sensor.

The logger unit collects data from the sensor network of the monitoring system, date-time stamps the data, and logs it into a database. The computer from each apartment is connected to a main storage server through a secure wired network connection. The main server receives, stores, and monitors sensor data of all apartments in TigerPlace for research purposes. Table 2-2 shows a snippet of the sensor firing data recorded in the log file for a resident (On October 5, 2005, around 12:30pm the person had several episodes of bed restlessness: 8 of 3-7 s, 2 of less than 3 s and one 7-15 s, and two episodes of low breathing). Note that the data in Table 2-2 is a sensor sequence of length 12.

Table 2-2. Sensor sequence snippet for a TigerPlace resident.

UserID	SensorID	Year	Month	Day	Hour	Minute	Second
3	3	2005	10	5	12	34	38
3	2	2005	10	5	12	36	52
3	2	2005	10	5	12	37	04
3	2	2005	10	5	12	37	11
3	1	2005	10	5	12	37	26
3	1	2005	10	5	12	37	28
3	2	2005	10	5	12	37	32
3	2	2005	10	5	12	41	18
3	2	2005	10	5	12	41	11
3	2	2005	10	5	12	41	4
3	5	2005	10	5	12	42	40
3	5	2005	10	5	12	42	58

The sensor network employs a variety of sensors to capture various aspects of the residents’ behavior. Figure 2-2 displays a floor map of a typical TigerPlace apartment that shows the placement of some of the sensors.

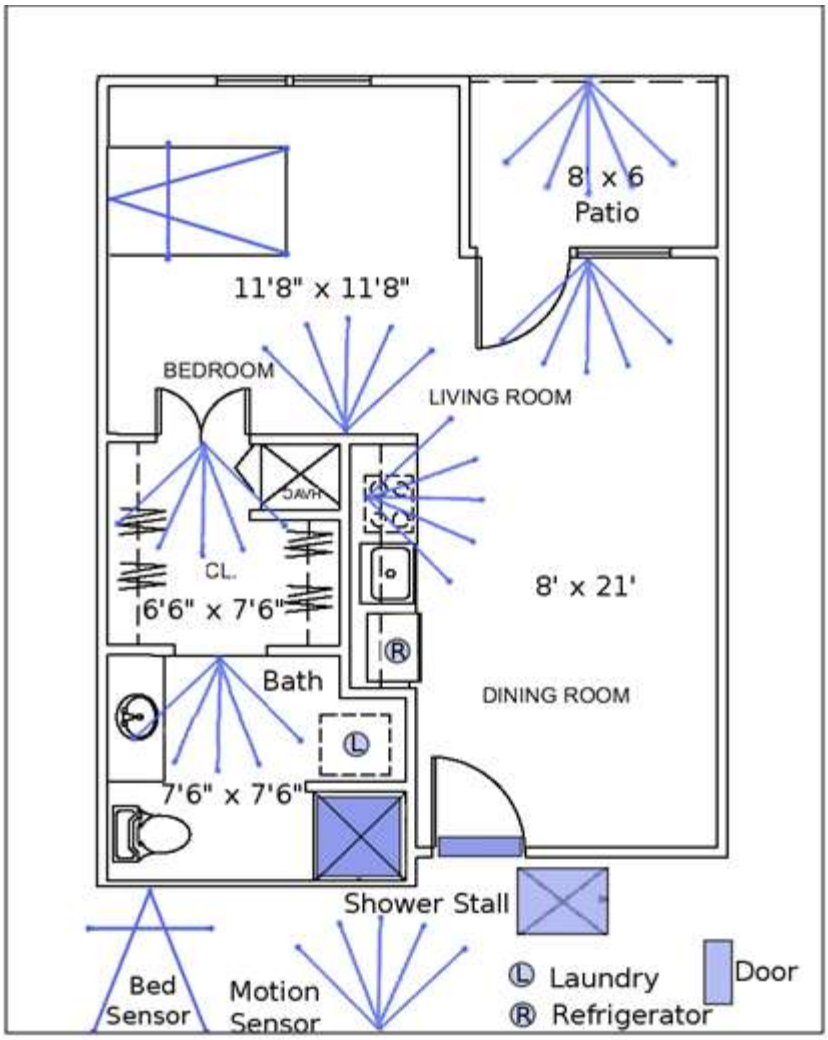


Figure 2-2. Typical TigerPlace apartment floor map with sensor deployment.

A web interface was developed to present the sensor data in a way that health care providers can easily understand and interpret. Users can filter their searches by resident information and a date interval. The sensors are grouped by type: motion, bed restlessness, bed pulse, and bed respiration. Histograms are used to display motion and bed sensor events, which are aggregated to a daily level. Figure 2-3 shows the user interface of the sensor display for health care providers and research staff. This interface shows aggregated motion sensors data from 2013-01-16 to 2013-01-30 for a resident.

Figure 2-3. Interface of the sensor display for health care providers and research staff [19].

2.2. Dataset

Table 2-3 describes the pilot sensor data from nine residents of TigerPlace used in this research with a total of 1902 sensor-days. For each resident, we also retrieved visit notes about various physical and/or mental health complains added by the nurses on site to the resident's EHR. In our dataset there are fewer notes than sensor data (automatically logged for each), as for some days there were no nursing comments. However, some residents (e.g. resident #3) may have multiple comments per day. In addition, each individual day was manually coded by the research team as normal or abnormal for each individual resident. Clinical experts retrospectively reviewed the TigerPlace EHR data (nursing comments, vital signs, etc.) to flag days (denoted as "abnormal") with resident illness episodes (column 3 in Table 2-3). We use the abnormal days to tune our sequence

similarity and test our early illness recognition algorithm. In the next section, we briefly present our preliminary experiments to health pattern detection.

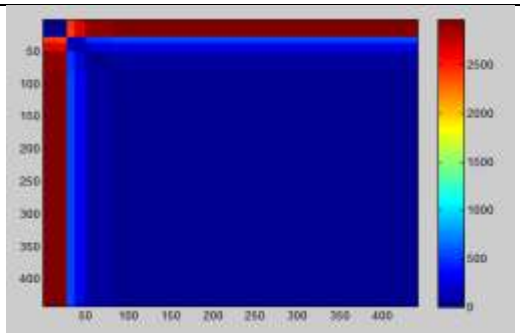
Table 2-3. Pilot dataset.

Resident#	Number of sensor days	Abnormal days
1	440	83
2	463	44
3	499	335
4	225	130
5	81	60
6	72	55
7	45	15
8	39	10
9	38	15

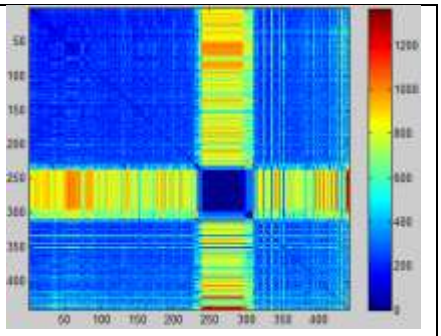
2.3. Data Visualization

To find the similarity of two sensor sequences many ways have been proposed and investigated in the literatures. One approach is to use Euclidean based distance functions to compute the distance between two sequences. In this approach, a sequence is considered as a point in an appropriate multi-dimensional space [20] [21] [22] [23] [24].

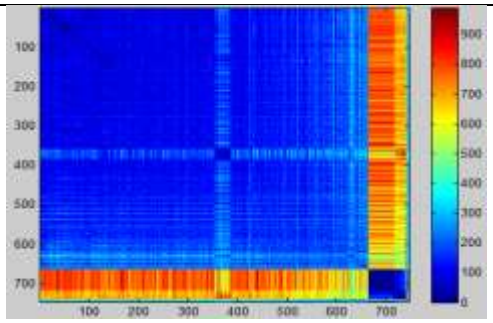
In our study, to predict abnormal days accurately, sensor data have been used to measure similar days in terms of patient's health status. However, considering each sensor data separately would not be helpful. As a proof, Figure 2-4 shows the results of iVAT algorithm [25] on the sensor data for resident #2 (see Table 2-3). The iVAT algorithm has been used to visualize the different possible clusters in the data [25]. The number of different diagonal blocks represents the number of possible clusters.



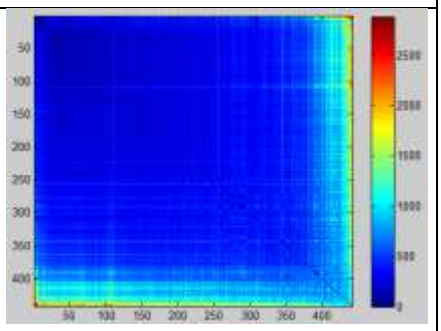
(a) Respiration sensor



(b)Breathing sensor



(c)Pulse sensor



(d)Motion sensor

Figure 2-4. Results of i-VAT algorithms for resident #2.

As Figure 2-4 shows, considering each sensor separately does not provide meaningful clusters of the sensor data. On the other hand, to predict abnormal days we should consider all aspects of activities of a resident captured by all sensors. To measure the similarity (or dissimilarity) of all sensor sequences, we use RMSE as a statistical measure of the magnitude of varying quantities which is widely used in signal processing and electrical engineering.

2.3.1. Sensor Data Conversion

To compare two days of sensor data with Euclidean based distances, we convert the sensor sequence to an aggregated time series. We aggregate sensor data time-wise (in one hour time intervals) and sensor type-wise (aggregated all motion sensor data together) obtaining a 3-dimensional sequence vector. For this experiment, we focus on sleep patterns. Therefore, we derive three dimensions as Plus (*P*), from sensor IDs 8,9,10, Breathing (*B*), sensor IDs 5,6,7, and Bed Movements (*M*), sensor IDs 1,2,3,4 from Table 2-1. For each dimension, we considered one hour time intervals, so the total number of related sensor hits per hour was counted to represent the point in the 3×24 dimensional

spaces. Let's denote the aggregated sensor hits on day X by $X=(P,B,M)$, where $P=(p_1, p_2, \ldots, p_{24})$, $B=(b_1, b_2, \ldots, b_{24})$, $M=(m_1, m_2, \ldots, m_{24})$. Figure 2-5 shows an example of the sequence of aggregated sensor data for resident #2 in January 1st, 2006.

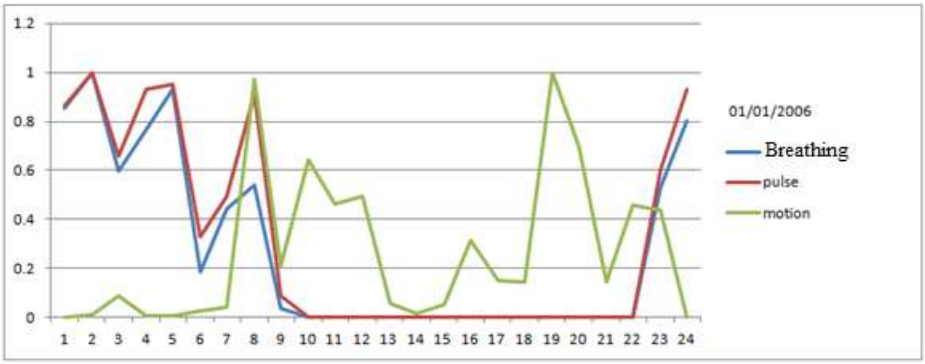


Figure 2-5. Aggregated sensor sequence for resident #2 at January 1st, 2006.

2.3.2. Euclidian Distance

The main goal is to predict patient's medical status based on the similar sequence data. To this goal, we have demonstrated our predicting system as follows. Sensor data have been aggregated using RMSE method. This technique has been used in complex wave forms made from common known wave forms. Consequently, this transferred data have been used to measure the similarity between sensor data of different days.

We used RMS approach, a widely used techniques in signal processing to find the distance between time series. That is, for two sensor sequences $X=\{x_{ij}\}$, $Y=\{y_{ij}\}$, $i \in \{1,2,3\}, j \in \{1,2,\ldots,24\}$, we can compute the RMS distance as:

$$\mathrm{RMS}(\mathrm{X},\mathrm{Y}) = \sqrt{\frac{1}{24}\left((\mathrm{X}_1-\mathrm{Y}_1)^2+\cdots+(\mathrm{X}_{24}-\mathrm{Y}_{24})^2\right)}, \qquad (2.1)$$

where $\mathrm{X_i} = \sqrt{\frac{1}{3}\left(p_{\mathrm{xi}}^2 + b_{\mathrm{xi}}^2 + m_{\mathrm{xi}}^2\right)}$, $\mathrm{Y_i} = \sqrt{\frac{1}{3}\left(p_{\mathrm{yi}}^2 + b_{\mathrm{yi}}^2 + m_{\mathrm{yi}}^2\right)}$.

To relate these data to staff's comments, the sensor data for those days that there are comments for have been selected. The staffs comment data is derived from the free text data which is parsed and converted to Lexicon Unique Identifier (LUI) and Concept Unique Identifier (CUI) using the UMLS. UMLS is a comprehensive list of biomedical terms that can be used in computer systems to understand the vocabulary of biomedicine and healthcare. The distances between comments have been measured using Jaccard distance [26].

To predict possible nursing notes given a sensor sequence data, we select LUI/CUI based on the distance of the RMS of the related sensor data. Using related sensor sequence, the set of close sensor's sequences based on the transferred data must have been found. Figure 2-6 shows our algorithm.

1. Given an unknown day *Y* and a training data set of size *n*, for each day:
    a. Aggregate sensor sequence and find its RMS with day *Y* using equation (2.1);
2. Normalize the RMS values using: $Normal_{RMS} = \frac{RMS - \min(RMS)}{\max(RMS) - \min(RMS)}$
3. Sort days based on their RMS values increasingly and select top 10 days;
4. Select the set of CUI's associated with selected tip 10 days;
5. For all selected CUIs
    a. If it is appeared in more than δ comment's of days, then add this CUI to the predicted CUI's for day *Y*.

Figure 2-6. Predicting comments algorithm based on sensor sequence similarity.

For this experiment, we used a sub set of our dataset, see Table 2-4. For three patients, the information from four sensors is gathered including motion, pulse, and breathing. Table 2-4 shows the amount of data and nursing notes for each resident. Based on the status of a resident in each day, different number of nursing comments has been placed where is shown in Table 2-4. We have tested this method on each resident data separately.

Table 2-4. Sample dataset for aggregated sensor data.

Resident#	Number of days	Number of nursing notes
1	440	83
2	463	44
3	499	335

Figure 2-7 shows an example of the sequence of three sensors for two specific dates for resident #2. As we can see in this Figure, analyzing each sensor data separately would not be helpful. However combining sensor data is a challenging task. The first graph shows the test sensor data which includes three sensors bed movements, pulse, and

breathing. And the second graph is the most similar day in terms of aggregated sensors to the test sensor data.

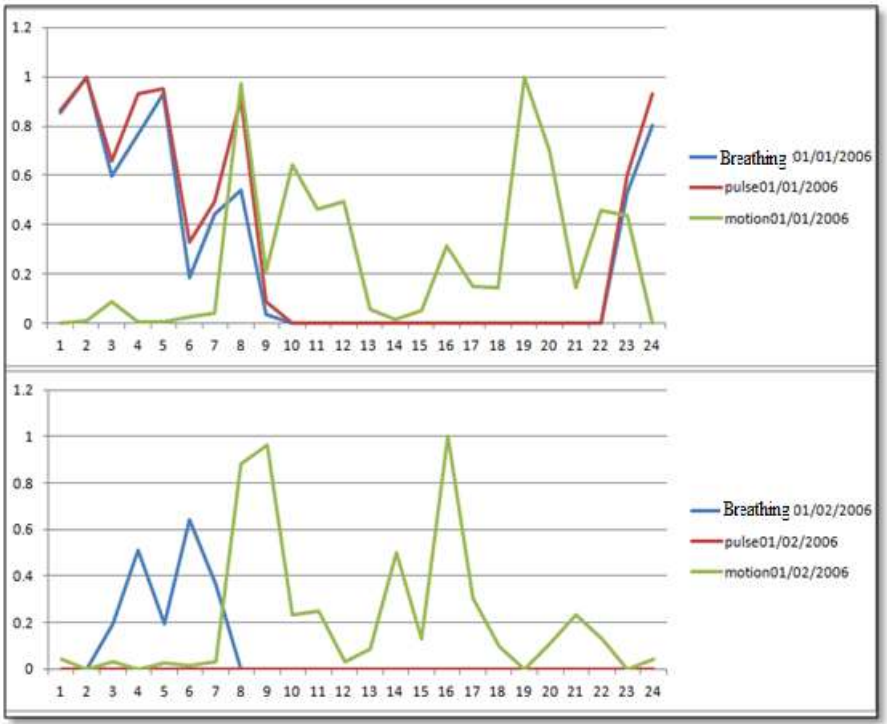


Figure 2-7. Sequence similarity for resident #2.

Table 2-5 demonstrates the results of our experiments on the detection of abnormal days. In this table we used *Precision*, *Recall* and *F_measure* as evaluation metrics. *F_measure* is the harmonic mean of *Precision* and *Recall*, where *Recall* is the true positive rate or sensitivity and *Precision* is the positive prediction value:

$$Precision = \frac{tp}{tp+fp} \tag{2.2}$$

$$Recall = \frac{tp}{tp+fn} \tag{2.3}$$

$$F\_measure = 2 * \frac{Precision*Recall}{Precision+Recall} \tag{2.4}$$

where *tp* is the number of true positives, *fn* is the number of false negatives, and *fp* is the number of false positives. For abnormal days detection, we use a *k*-nearest neighbor classifier (*k*=7) using RMS as similarity function with a leave-one-out cross-validation approach.

Table 2-5. Similarity results using Euclidean distance.

Resident#	Precision	Recall	F-measure	Number of Days

1	0.4256	0.7446	0.5388	283
2	0.2897	0.3559	0.3174	22
3	0.3361	0.4639	0.3825	53

Euclidean distance does not perform very well in terms of F-measure on detecting abnormal days. Euclidian distance is sensitive to outliers. Moreover, it does not capture the time between events and cannot be computed for sequences of variable length. The main limitation of using the Euclidean distance as a similarity measure is due to its weak performance in the presence of noise. The second commonly used approach to MATS similarity is based on the non-Euclidean metrics, such as Dynamic Time Warping (DTW) [27].

2.3.3. DTW Distance

DTW is a well-known method that measures the similarity of two time series by alignment. DTW finds an optimal alignment between two given time series (sequences) with certain restrictions (see Figure 2-8) [28]. In this method, sequences are warped nonlinearly to match each other. DTW has been used widely in speech recognition to detect different speech patterns [28]. We will continue this chapter with briefly discussing the main idea of DTW and applying this method on our dataset for the detection of abnormal days using sensor sequences.

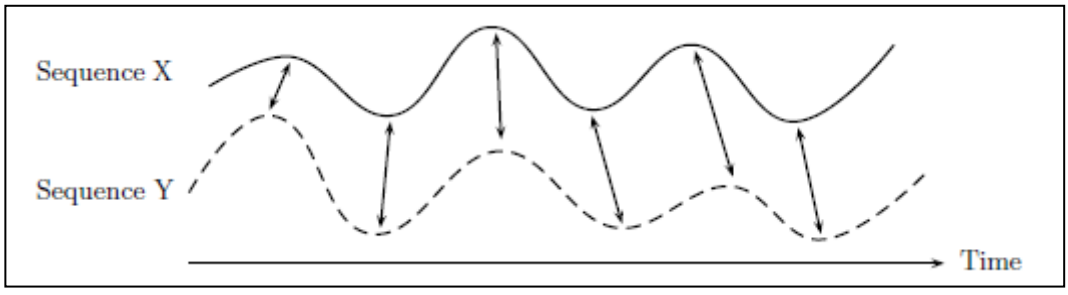


Figure 2-8. DTW for aligning two sequences. Aligned points are shown by the arrows.

2.3.3.1. DTW Definition

The aim of DTW is to compare two time dependent sequences as $X = (x_1, x_2, \ldots, x_n)$ of length n and $Y=(y_1, y_2, \ldots, y_m)$ of length m where $n, m \in \mathbb{N}$. Sequences X and Y may be discrete time-series or, more generally, a sequence of features that are sampled at equidistant points in time. Let's denote the feature space by F. Then $x_n, y_m \in F$ for $n \in [1:N]$ and $m \in [1:M]$. We need a local cost measure to compare two different features $x, y \in F$. The cost measure is also known as local distance measure, which is defined as a function $c: F \times F \rightarrow R \geq 0$. Typically, $c(x, y)$ is small (low cost) if x and y are similar to each other, and otherwise $c(x, y)$ is large (high cost) [28].

We define a cost matrix as $C \in \mathbb{R}^{N \times M}, C(n,m) = c(x_n, y_m)$, to evaluate the local cost measure for each pair of elements of two sequences X and Y. The task is to find an alignment between X and Y that minimized the overall cost. Intuitively, such an optimal alignment runs along a "valley" of low cost within the cost matrix C. Figure 2-9 illustrates this method.

Algorithm: **Dynamic Time Warping**

Input: Accumulated cost matrix C

Output: Optimal warping path p*.

Procedure: The optimal path $p^* = (p_1, \ldots, p_L)$ is computed in reverse order of the indices starting with $p_L = (N,M)$. Suppose $p_l = (n,m)$ has been computed. In case $(n,m) = (1, 1)$, one must have $l = 1$ and we are finished. Otherwise,

$$p_{l-1} = \begin{cases} (1, m-1) & if\ n = 1 \\ (n-1, 1) & if\ m = 1 \\ arg_{min}\{C(n-1, m-1), C(n-1, m), C(n, m-1)\}, & otherwise \end{cases}$$

where we take the lexicographically smallest pair in case "arg_{min}" is not unique.

Figure 2-9. DTW algorithm.

In [27], authors rephrased the problem of finding similar time series instead of a problem of sequence alignment. They presented an algorithm for DTW on multi-attribute time series and compared it to the ordinary DTW, where a single attribute is used for sequence alignment. The DTW algorithm for two sequences calculates the distance between each possible pair of data points. Then, the algorithm uses these distances to calculate a cumulative distance matrix, and it finds the best path through this matrix.

In another approach, the problem of finding similar time sequences is divided into whole matching and subsequence matching. If the lengths of two sequences are similar, it is called whole matching; otherwise, it is subsequence matching. Discrete Fourier

Transform (DFT) has been used to transform the time sequences to the frequency domain. Haar wavelet transform have been used to reduce the dimensionality of the time sequences [29]. The shift and scale comparisons of the sequences have been used in another study to allow comparisons under different experimental conditions [30]. The result of applying original DTW method as the similarity measure on our dataset has the average F-measure of 35% on dataset presented in Table 2-4. DTW has also weak performance on MATS problem on our dataset.

2.4. Conclusion

In this chapter, we presented the system architecture at TigerPlace. We formulate the problem of EIR using MATS similarity on sensor sequences collected from in-home sensor network. We reviewed two similarity measures (Euclidean based and non-Euclidean based methods) that have been used to predict abnormal days on TigerPlace dataset and discussed their performance. These methods require data aggregation on sensor time series which affects the performance of prediction. In the next chapter, we review related literatures in time series data mining. We investigate different approaches on time series data representation and continue with discussing several methods on supervised and unsupervised learning on time series. We also provide a review on well-known similarity measures for MATS applications.

Chapter 3 Literature Review

By 2030, 20% of America's population will be 65 or older, growing from 35 million in 2010 to 71.5 million in 2030 [31]. Most of seniors suffer from chronic conditions such that more than 40% of them need assistant in performing activity of daily livings. Using technology the early illness recognition and interventions can address the needs of older adults while minimizing the cost of hospitalization [32].

In the US, among 10 million people who need long term care, 46% of them are older adults. If all of them have the opportunity to use the early illness recognition framework developed at TigerPlace, there is a potential of $89 billion in cost savings. That is more than 40% of all dollars spent on people with long-term health needs in the US. Technology coupled with nurse care coordination has huge potential to help older people stay at home, where they want to be, safely and more cost-effectively [32].

In this chapter, we discuss existing elderly smart homes and provide a literature review on methods and techniques used for early illness recognition.

3.1. Health Monitoring System for Eldercare

Eldercare has become more challenging as older adults prefer to live independently for as long as they are able regardless of their conditions such as frailty, dementia, etc. Late health assessment is another risk factor for an older adult that usually occurs because of some factors such as impressing health changes as aging effects, fear of being institutionalized, and the failure of physician's assessments [5].

Using sensor networks for health care monitoring systems patterns of activities can be detected and medication compliance can be assessed using a collection of sensors and predictive algorithms [33] [34]. However, considering the health context of the monitored patient is still an unsolved problem.

Health context of each individual is a very important factor to be considered, since normal behaviors can change because of a hospitalization, whereas sleep patterns or fall risks can be affected by changing in medication. Sensor data and contextual health information such as chronic conditions and practitioners comments provided by EHR and

Tele health data have been used to identify health patterns [35]. Sensor data have been used to identify early signs of illnesses. Based on changing sensor patterns, alerts are sent to clinical staffs that provide feedbacks on the clinical relevance of each alert. To detect changes in sensor patterns, defining suitable similarity function for sensor sequences is a crucial task.

One efficient approach to health monitoring is to use sensor networks for collecting information about the older adult's activities. In the last decade, many of such health monitoring systems have been piloted. For example, MIT's PlaceLab, Georgia Tech's aware House, Honeywell's Independent Lifestyle Assistant, and University of Missouri' s TigerPlace have demonstrated possible approaches for activity monitoring [6] [7] [8] [9]. A variety of methodologies for activity detection and assessing medication compliance have been reported in the literature [10] [11] [36]. Even though these and many other systems are successful examples of applying sensor networks to monitor activity patterns, the major unsolved challenge is to consider the health context of the monitored older adults.

To understand the predicted event in an EIR framework, it is required to interpret data from transmitted sensor data with feature extraction. As a preprocessing step for analyzing a huge volume of sensor data, most environmental monitoring applications need a well-organized data representation model with rapid data processing which abstracts local data.

3.2. Time Series Data Representation

Collected data from sensor networks are represented by categorical time series. Representing categorical time series data is a fundamental task in time series data mining. In categorical time series, each data point is a symbol that represents a category where it belongs to or in our case a sensor, whereas in non categorical time series each data point is a number. Since our data set is an example of categorical time series data our focus is more on publications related to this field, while we briefly review other approaches as well.

An important task in preprocessing of time series data is dimensionality reduction. Difficulties in searching and mining time series data arise from the high dimensionality

and the sheer size of data. A common approach for dimensionality reduction is to first transform data into another domain and next apply an indexing mechanism. For various time series data mining tasks such as segmentation, clustering, and classification, measuring the similarity between two time series or time series sub sequences is a core task. Depending on the time series representation, different mining tasks can be roughly classified into four fields including: pattern discovery and clustering, classification, rule discovery and summarization. Data series representation is also important in the dimensionality reduction of data. There are two approaches to design an abstraction model of time series data including: aggregated data approach, and non aggregated data approach.

3.2.1. Aggregated Data Approach

One simple approach to represent time series is in the form of sampled time series with a specific sampling rate. In this method, the rate of sampling should be low and missing some values should not raise a problem. An improved version of this method is called Piecewise Aggregate Approximation (PAA) that uses the mean value of each segment to represent the time series [37]. Keogh et al. extended this method in [38] and name it as Adaptive Piecewise Constant Approximation (APCA). In this method, the length of each segment is not fixed but totally versatile and adaptive to the original time series data. Another representation time series data method that uses the average value for each equal sized segment using PAA approach was reported in [39]. This method that originally called "piecewise constant approximation" uses the segmented means to represent time series data. Figure 3-1 shows the outline of the indexed searching algorithm. In this figure, $X = x_1 \ldots x_n$ and $Y = y_1 \ldots y_m$ are two time series. And time series X of length n is represented in N space by a vector $\bar{X} = \overline{x_1}, \ldots, \overline{x_N}$.

```
best-so-far ← infinity;
done ← FALSE;
i ← 1;
X̄ ← transformed(X); // Using D_true(A,B) >= D_index space(A,B)
while i ≤ G AND NOT(done)
        Find X̄ 's i^th nearest neighbor in the index; // Using x̄_ι = (N/n) Σ_{j=n/N(i-1)+1}^{n/N i} x_j
        Retrieve sequence represented by the i^th nearest neighbor;
        if D(original-sequence_i,X) < best-so-far //D(X,Y) = √(Σ_{i=1}^{n}(x_i − y_i)^2)
                best-so-far ← D(original-sequencei,X);
        end;
        if best-so-far ≤ i^th+1 nearest neighbor in the index
                done ← TRUE;
                Display('Sequence ',i,' is the nearest neighbor to Query');
                Display('At a distance of ',best-so-far );
        end;
        i ← i + 1;
end;
```

Figure 3-1. An outline of the APCA indexing searching algorithm.

Another well-known group of time series representation approaches converts the numeric time series to symbolic form. In fact, the time series is first discretized into segments, then each segment will be converted into a symbol [40] [41] [42]. Symbolic Aggregated approXimation (SAX) is a method that first proposed by Lin in 2003 to convert the result from PAA to a symbolic string [43] [44]. The *y*-axis of the time series (distribution space) is divided into regions with the same probability distributions. Then each region is mapped into a symbol. Finally, the transformed time series $\hat{T}$ is converted into a string of symbols ($S_1, S_2, ..., S_n$) using PAA approach. Figure 3-2 (left) shows a short time series *T* and the right plot shows this time series converted into PAA representation. PAA maps a time series *T* of length *n* into a *w*-dimensional space and represents the time series by a vector of real numbers $\bar{T} = \bar{t}_1, \ldots, \bar{t}_w$ where the *i*th element of $\bar{T}$ is calculated using equation (3.1) [44].

$$\bar{t}_i = \frac{w}{n} \sum_{j=\frac{n}{w(i-1)+1}}^{\frac{n}{w}i} T_j \qquad (3.1)$$

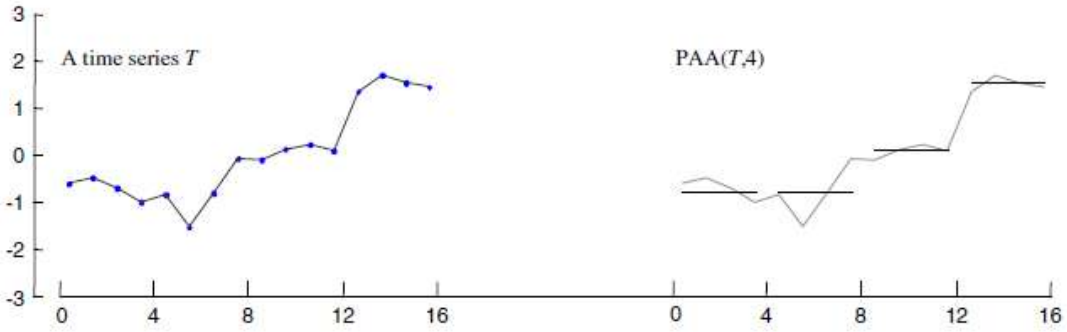


Figure 3-2. (left) A time series T, of length 16. (right) A PAA approximation of T, with 4 segments [44]

In this method, there are two parameters that need to be initialized including the length of subsequence and the size of alphabet that is the number of symbols used for conversion. The SAX representation takes the PAA representation as an input and discretizes it into a small alphabet of symbols with a cardinality of size *a*. The discretization is achieved by imagining a series of breakpoints running parallel to the *x*-axis and labeling each region between the breakpoints with a discrete label. Any PAA value that falls within that region can then be mapped to an appropriate discrete value. To have uniform symbol distribution in a SAX word, a sorted list of numbers should be used such that with *Breakpoints* $=\beta_1,\ldots,\beta_{a-1}$, the area under a *N(0,1)* Gaussian curve from β_i to $\beta_{i+1}=1/a$ (β_0 and β_a are defined as $-\infty$ and ∞, respectively). Then, A SAX word is simply a vector of discrete symbols as shown below such that the cardinality of the SAX word is represented as superscript.

$$\text{SAX}(T,w,a) = \boldsymbol{T}^{\boldsymbol{a}} = \{t_1, t_2, \ldots, t_{w-1}, t_w\}$$

Each SAX symbol can be represented as a letter, integer, or the binary code of an integer. Figure 3-3 shows two SAX words of a time series *T* of length 16. The length of word in both examples is 4, but one has a cardinality of 4 and the other has a cardinality of 2. Therefore, the SAX words T^4 and T^2 can be represented as following:

$\text{SAX}(T, 4, 4) = T^4 = \{11, 11, 01, 00\}$

$\text{SAX}(T, 4, 2) = T^2 = \{1, 1, 0, 0\}$.

In this presentation once the word T^2 can be derived from the word T^4 simply by ignoring the trailing bits in each of the four symbols in the SAX word.

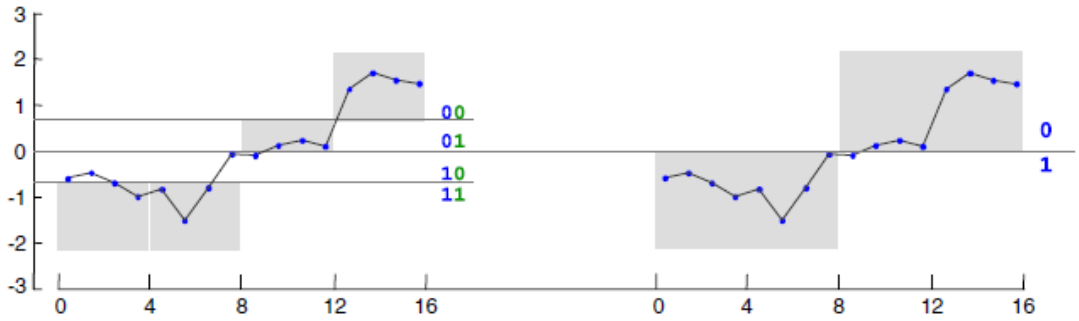


Figure 3-3. A time series *T* converted into SAX words of cardinality 4 {11, 11, 01, 00} (left), and cardinality 2 [44]

By representing time series in SAX words, time series can be converted to any lower resolution that differs by a power of two, simply by ignoring the correct number of bits.

In [45] authors proposed to use each segment by a codeword from a codebook of key-sequences. Then they extended this method to multi-resolution consideration. To consider the temporal order of values, Morchen and Ultsch [46] proposed an unsupervised discretization process, called Persist algorithm, based on quality score and persisting states. This algorithm incorporates temporal information of time series in the discretization process.

3.2.2. Non Aggregated Data Approach

In applications such as activity recognition in large and complex datasets, data is represented in a non aggregated format. In this approach, extracting activity information from time series data is still an open problem in applications such as video understanding, activity monitoring for healthcare and surveillance. In a novel approach, the Frequent Temporal Pattern (FTP) algorithm was first introduced by Magnusson in [47] to extract frequent human behaviors and improved in [48] for artificial sensor data. Temporal patterns (T-patterns) are represented by symbols and time stamps. FTP algorithm finds possible relationships between pairs of symbols by building trees of temporal dependencies. We review this method in more details in chapter 9.

Most publications in activity recognition from non aggregated data can be divided into two approaches: Bag of Words (BoW) approach, and Bioinformatics approach.

3.2.2.1. Bag of Word Approach

In 1971, the BoW approach was introduced to retrieve information from text [49]. Researchers have applied this approach extensively for text analysis, indexing and retrieval [50]. Building on the success of BoW approaches for Information Retrieval (IR) with text and images, researches in activity recognition have focused on working with BoW using local spatio-temporal features [51]. More recent publications apply BoW approach with robust descriptors, which exploit continuous object motion and integrate it with distinctive appearance features [52]. BoW approach using features based on dense trajectories [53] and features learnt in an unsupervised manner directly from video data [54] are also investigated. However, when activities are represented as bags of words, the underlying sequential information provided by the ordering of the words is typically lost. To address this problem, *n*-grams approaches have been used to retain some of the ordering by forming sub-sequences of *n* items [50]. More recently, variants of the *n*-gram approach have been used to represent activities in terms of their local events sub-sequences [55].

3.2.2.2. Bioinformatics Approach

During the analysis of activity recognition, the task of assessing the similarity of behaviors can be cast as computing the similarity of multiple MATS. MATS similarity is a challenging task and is still an open research question. Our novel approach to sequence similarity is to consider MATS as a one-dimensional sequence and to use bioinformatics techniques to find the best alignment. This approach is possible when sensor data is already quantized by the hardware system, which results in a symbolic sequence of sensor firings and the related firing time. SW algorithm is a well-known algorithm to find an optimal local alignment [14]. Researchers have used SW not only in bioinformatics, but also in natural language processing [56] [57]. In [57], authors aligned words using SW without a pre-existing ontology. In [29], authors implemented a search tool on protein structures by integrating SW with fuzzy logic to match protein energy profiles.

The idea behind SW algorithm is to use dynamic programming to find the optimal local alignment with respect to a pre-defined scoring system. The scoring system includes a substitution matrix and a gap-scoring scheme. SW algorithm calculates a local similarity score in linear space. In bioinformatics, one motivation for the local alignment is in situations where biological sequences are highly related to each other, but the similarities in local regions are very low. In this case, it is hard to find the correct alignment. To tackle this problem, a local alignment discards such regions and concentrates on those with high similarities. Another motivation for local alignment is to use a reliable statistical model for optimal local alignment to produce expectation values [30].

SW method can be used in time series similarity to find best local matches. In fact, finding local similarity in sensor time series is desirable, since it shows similar health patterns in the local sequence regions. However, for motion sensor data, the time stamp is an important factor that needs to be considered when finding optimal local time series alignments. Unfortunately, the classical SW algorithm [14] does not consider time in alignment.

3.3. Time Series Data Analysis

Data mining analysis on time series has been explored by researches in many domains and applications. Various data mining tasks on time series analysis can be divided into four categories including [58]:

1) **Indexing** (Query by Content): Given a query time series *Q*, and some similarity/dissimilarity measure *D(Q,C)*, find the nearest matching time series in database *DB*.
2) **Clustering**: Find natural groupings of the time series in database *DB* under some similarity/dissimilarity measure *D(Q,C)*.
3) **Classification**: Given an unlabeled time series *Q*, assign it to one of two or more predefined classes.
4) **Segmentation**: Given a time series *Q* containing *n* data points, construct a model *Q'*, from *K* piecewise segments ($K << n$) such that *Q'* closely approximates *Q*.

The segmentation task is important for two major uses. First, it is used to determine the change in the underling time series model [59] [60]. Second it is performed to create a high level representation of the time series for other tasks such as indexing, clustering, and classification [60] [61].

Time series analysis has been of interest of statisticians. Modeling categorical time series analysis is still an open question in this field. In the next two sections, we review selected publications in both fields of statistics and machine learning research community.

3.3.1. Statistical Approach in Categorical Time Series Analysis

In the statistics literature, there is a long list of articles that represents different strategies that have been proposed for modeling of categorical time series. The most common research questions are:

1. Is there an apparent "periodic" tendency in the data?
2. What is the best way to predict a future state of the variable of the time series?
3. Do lagged values of the state of the variable of the time series determine future states?

4. Can the covariates be used to predict the state of the variable of the time series?

The most common approach to answer such questions is by considering regression models for categorical time series.

A categorical time series is represented as following. The observed categorical time series is shown by $\{Y_t\}$, $t = 1, \ldots, N$. In other words, for each t , the possible values of Y_t are $1, 2, \ldots, m-1, m$, where m is the number of categories, and the "first" category is assigned the integer value of 1, the "second" category is assigned the integer value of 2 and so on. The assignment of integer values to the categories is a matter of convenience; hence, it is not unique [62]. Assigning integer values to categories introduces arbitrariness in data. To reduce this side effect, the t^{th} observation of categorical time series is represented by the vector $Y_t = (Y_{t1}, \ldots, Y_{tq})'$ where $q=m-1$ is the length of vector with elements

$$Y_{tj} = \begin{cases} 1, & \text{if the } j\text{th category is observed at time } t \\ 0, & \text{otherwise,} \end{cases} \tag{3.1}$$

for $t = 1, \ldots, N$ and $j = 1, \ldots, q$. The vector of transitional probabilities is denoted by $\pi_t = (\pi_{t1}, \ldots, \pi_{tq})'$.

Based on the theory of generalized linear models, McCullagh and Nelder [63] assume a link between that the vector of transition probabilities—that is, the conditional expectation of the response vector— and the covariate process. They show this connection using the following equation

$$\pi_t(\beta) = h(Z'_{t-1}\beta), \tag{3.2}$$

where $\boldsymbol{\beta}$ is a p-dimensional vector of time-invariant parameters. Equation (3.2) gives the general form of a multivariate generalized linear model for categorical time series. This model has been considered by researches such as Fahrmeir and Kaufmann [64], Kaufmann [65], Fokianos et. al. [66], Brillinger [67], Agresti [68], etc.

The major statistical question in a regression model for categorical time series is to estimate the vector of parameters β. Cox [69] proposed the partial likelihood method to solve this problem. This method uses the martingale (probability) theory [70] to approach the problem of estimating and testing the vector of parameters.

Fokianos and Kedem [62] applied the regression theory by generalizing linear model and partial likelihood to model categorical time series. They used a variety of models to illustrate the selection of the link function. They showed that regression

methods for categorical time series allow for parsimonious modeling and incorporation of random time-dependent covariates. They specifically analyzed nominal and ordinal time series and compared to Markov chains.

Regression analysis is a statistical process for estimating the relationships among variables. In time series analysis, linear regression is a basic technique to find dominant trends or comparing time series to find similar or dissimilar curves. A linear fit for time series $y(t)$: $t\epsilon[t_s,t_e]$ is a linear estimation function: $\hat{y}(t) = \hat{\theta} + \hat{\gamma}t$ where $\hat{y}(t)$ is the estimated value of the $y(t)$, and γ, and Θ are two parameters. In this notation, the difference $y(t) - \hat{y}(t)$ is the residual for time t.

Chen et. al. [71] proposed a multi-dimensional regression analysis of time-series stream data that only uses a small number of compressed stream data for regression analysis. Stream data is generated continuously in a dynamic environment, with huge volume, infinite flow, and fast changing behavior. They used regression as measure in the data cube model, as defined in online analytical processing (OLAP) systems, to minimize the amount of data that needed to be retained in the memory for analysis. In the proposed method, stream time series data, data cube, is compute and store only two critical layers (which are essentially cuboids) in the cube: (1) an observation layer, called *o-layer*, which is the layer that an analyst (or the system) checks and makes decisions for either signaling the exceptions, or drilling on the exception cells down to lower layers to find their corresponding exception "supporters"; and (2) the minimal interesting layer, called *m-layer*, which is the minimal layer that an analyst would like to study, since it is often neither cost effective nor practically interesting to examine the minute detail of stream data.

Challenges in regression analysis on categorical time series data are ergodicity, stationary, and maximum likelihood estimations. Researchers have addressed these challenges in literatures [72] [73] [74]. Ergodicity describes a random process for which the time average of one sequence of events is the same as the ensemble average. Stationary represent a characteristics of a time series or a stochastic process that its joint probability distribution does not change when shifted in time. That means parameters such as the mean and variance do not change over time and do not follow any trends. Modeling non-stationery time series is still an open question. Ahmad et al. [75] presented

a multi scale wavelet analysis on non-stationary, volatile and high- frequency time series data. This method separates the trend, cyclical fluctuations, and autocorrelation effects. The proposed method uses Discrete Wavelet Transform (DWT) to decompose a time series into several sub-series. The DWT of a time series X is obtained using a series of filtering calculations. In the first step, a low pass filter with impulse response g is applied. Then, using a high-pass filter h the signal will be decomposed simultaneously.

$$y[n] = (x * g)[n] = \sum_{k=-\infty}^{\infty} x[k]g[n-k] \quad (3.3)$$

$$y_{low}[n] = \sum_{k=-\infty}^{\infty} x[k]h[2n-k] \quad (3.4)$$

$$y_{high}[n] = \sum_{k=-\infty}^{\infty} x[k]g[2n-k] \quad (3.5)$$

Applying DWT filtering process recursively on time series the long and shorter fluctuations will be removed from the signal. Therefore, a linear trend can be achieved. Each recursive component is a time series in its own right.

3.3.2. Machine Learning Approach in Categorical Time Series Analysis

In the machine learning community, researches proposed different methods for time series analysis depending on the dataset, available information, and research questions. In this section, we review those approaches very briefly and in the following sections describe selected methods in more details. In general, the machine learning publications in time series analysis can be divided into two groups of supervised learning, and unsupervised learning. If the labels of data points are available, then supervised learning approaches are used to mainly train a classifier for the purpose of prediction. Whereas, in the absence of labeled data unsupervised learning approaches are used to cluster dataset and find boundaries between them.

Supervised learning is a machine learning task that uses labeled data to infer the possible nonlinear dependence between historical data and future data. Classification is a supervised learning task that learns a function from training data to assign a label to the test data. Formally, given a set of N labeled training data of the form $\{(x_1,y_1), \ldots, (x_N,y_N)\}$ such that x_i is the feature vector of the i^{th} example and y_i is its label (or its class), a supervised learning method finds a function $h: X \rightarrow Y$ in the input space X to the output space Y.

Classification is a traditional data mining task. However, in time series domain this task is more challenging due to complication of the data. In literature, researchers addressed the classification of time series data by proposing new methods such as combining local properties [122], wavelet decomposition [123], meta-feature approach like local maxima in time series [124], feature subset selection (FSS) [125], Gaussian Mixture model [126], and many more. In this section, we review the most common approaches in classification task and their performances on time series classification.

Moreover, a huge amount of publications in this area introduce new similarity measures that can be used in the variety of machine learning tasks. The similarity measures are important in indexing and segmentation tasks. In the following sections we discuss these subjects in more details and review related literatures.

3.4. Indexing and Similarity Measures

Indexing and similarity measure are of fundamental tasks in time series analysis and data mining. In dimensionality reduction of time series data indexing is a powerful tool. Moreover, researchers incorporated different indexing methods for time series representation. Basic approaches in indexing time series are proposed by Guttman [76] in 1984 who used R-tree to develop a multidimensional indexing structure, and Agrawal [77] in 1993 who adopted R*-tree and proposed F-index method to index the first few Discrete Fourier Transform (DFT) coefficients. One year later, in 1994, ST-index was proposed by Faloutsos to extent the previous works [78].

For indexing time series that are represented by Piecewise Constant Approximation (PCA), authors proposed a multi-level distance based index structure [79]. In [80] a hybrid indexing structure on Euclidean and periodic space was developed and named multi-metric (MM) tree. This was an extension of their previous work that uses Minimum Bounding Rectangle (MBR) for indexing [81]. This index was developed based on the Fourier transform and a multi-resolution index was build based on the wavelet transform. To index massive collection of time series, *i*ndexable *S*ymbolic *A*ggregate appro*X*imation, *i*SAX, is proposed based on SAX [82]. Since the main bottleneck in mining massive time series data is the time needed for building an index, *i*SAX is proposed to make fast indexing possible by providing zero overlap at leaf nodes.

*i*SAX uses the same presentation as SAX, that is a SAX word. Symbols in a SAX word can be binary digits. However, since it is tedious to write binary digits, the *i*SAX representation uses integer numbers with a superscript. The superscript indicates the cardinality of *i*SAX words to avoid confusion. For example, a time series *T* can be represented by an *i*SAX word as:

$$iSAX(T, 4, 8) = \boldsymbol{T^8} = \{6^8, 6^8, 3^8, 0^8\}$$

Because the individual symbols are ordinal, exponentiation is not defined for them, so there is no confusion in using superscripts in this context. Two *i*SAX words, *T* and *S*, with different cardinalities can be compared as following:

$$iSAX(T, 4, 8) = T^8 = \{110, 110, 011, 000\} = \{6^8, 6^8, 3^8, 0^8\}$$
$$iSAX(S, 4, 2) = S^2 = \{0, 0, 1, 1\} = \{0^2, 0^2, 1^2, 1^2\}$$

To compare *T* and *S*, first the lower cardinality representation is *promoted* into the cardinality of the larger, and then two *i*SAX words of the same cardinality are giving it to the *MINDIST* function as shown in equation (3.6).

$$MINDIST(T,S) \equiv \sqrt{\frac{n}{w}}\sqrt{\sum_{i=1}^{w}(dist(t_i, s_i))^2} \qquad (3.6)$$

The Euclidean distance can be used as a *dist* function. This *i*SAX word provides an admissible lower bound. The use of *i*SAX allows for the creation of index structures that are hierarchical, containing non-overlapping regions (unlike R-trees etc.), and a controlled fan-out rate. A set of time series represented by an *i*SAX word can be split into two mutually exclusive subsets by increasing the cardinality along one or more dimensions. The number of dimensions *d* and word length, *w*, $1 \leq d \leq w$, provide an upper bound on the fan-out rate. If each increase in cardinality per dimension follows the assumption of iterative doubling, then the alignment of breakpoints contains overlaps in such a way that hierarchical containment is preserved between the common *i*SAX word and the set of *i*SAX words at the finer granularity. Specifically, in iterative doubling, the cardinality to be used after the *i*th increase in granularity is in accordance with the following sequence, given base cardinality *b*:$b*2^i$. The maximum fan-out rate under such an assumption is 2^d.

The index is constructed given base cardinality *b*, word length *w*, and threshold *th* (*b* is optional; it can be defaulted to 2 or be set for evaluation to begin at higher

cardinality). The index structure hierarchically subdivides the SAX space, resulting in differentiation between time series entries until the number of entries in each subspace falls below *th*. Such a construct is implemented using a tree, where each node represents a subset of the SAX space such that this space is a superset of the SAX space formed by the union of its descendents. A node's representative SAX space is congruent with an *i*SAX word and evaluation between nodes or time series is done through comparison of *i*SAX words. Figure 3-4 shows the pseudo-code of the insert function used for index construction.

```
1.Function Insert(ts)
2.   iSAX_word = iSAX(ts, this.parameters)
3.   if Hash.ContainsKey(iSAX_word)
4.         node = Hash.ReturnNode(iSAX_word)
5.         if node is terminal
6.               if SplitNode() == false
7.                     node.Insert(ts)
8.               else
9.                     newnode = new internal
10.                    newnode.Insert(ts)
11.                    foreach ts in node
12.                          newnode.Insert(ts)
13.                    end
14.                    Hash.Remove(iSAX_word, node)
15.                    Hash.Add(iSAX_word, newnode)
16.              endif
17.        elseif node is internal
18.              node.Insert(ts)
19.        endif
20.  else
21.        newnode = new terminal
22.        newnode.Insert(ts)
23.        Hash.Add(iSAX_word, newnode)
24.  endif
```

Figure 3-4. iSAX index insertion function

Given a time series to insert, first the *i*SAX word representation are obtained using the respective *i*SAX parameters at the current node (line 2). If the hash table does not yet contain an entry for the *i*SAX word, a terminal node is created to represent the relevant SAX space, and the time series is inserted accordingly (lines 22–24). A terminal node is a leaf node which contains a pointer to an index file on disk with raw time series entries. If the hash table is not empty, and there is an entry in the hash table, then the corresponding node is fetched.

If this node is an internal node that is designated to split a SAX space and is created when the number of time series in a terminal node exceeds threshold *th*, then its insert function will be called recursively (line 19). If the node is a terminal node, occupancy is evaluated to determine if an additional insert warrants a split (line 7). If so, a new internal node is created, and all entries enclosed by the overfilled terminal node are inserted (lines 10–16). Otherwise, there is sufficient space and the entry is simply added to the terminal node (line 8).

In large time series data sets an indexing scheme provides an efficient organization of data for quick retrieval. Most of the researches in this area involve a dimensionality reduction for indexing using a spatial access method. The differences of these methods are less on indexing power and more on the quality of results and the retrieval speed [83]. Two main issues in designing an indexing schema are *completeness* (no false dismissals) and *soundness* (no false alarms). The main properties of an indexing method are summarized below [84]:

(i) It should be much faster than sequential scanning.

(ii) The method should require little space overhead.

(iii) The method should be able to handle queries of various lengths.

(iv) The method should allow insertions and deletions without rebuilding the index.

(v) It should be correct, that is, there should be no false dismissals.

Similarity between time series is an essential sub routine in almost every time series mining task. The broadly used Euclidean distance is unable to reach levels of abstraction in characteristics of time series that human can reach such as amplitudes, scaling, temporal warping, noise, and outliers. Numerous authors have mentioned several drawbacks of using L_p norms [85] [58]. Many authors have proposed different transformations similarity for a time series $T=\{t_1, t_2, \ldots, t_n\}$ of n data points such as:

- Amplitude shifting: This is the series $S=\{s_1, \ldots, s_n\}$ obtained by a linear amplitude shift of the original series $s_i=t_i+k$ with $k \in \mathbb{R}$ a constant [86].
- Uniform Amplification: The uniform amplification is the series S obtained by multiplying the amplitude of the original series $s_i=k.t_i$ with $k \in \mathbb{R}$ a constant [86].

- Uniform time scaling: This is the series $S=\{s_1, \ldots, s_n\}$ produced by a uniform change of the time scale of the original series $s_i=t_{[k,i]}$ with $k \in \mathbb{R}$ a constant [87].
- Dynamic amplification: This is the series S obtained by multiplying the original series by a dynamic amplification function $s_i=h(i).t_i$ with $h(i)$ a function such that $\forall t \in [1 \ldots n], h'(t) = 0 \; if \; and \; only \; if \; t'_i = 0$ [88].

A similarity measure should be robust to any combination of transformations. Similarity measure can be divided into four categories including:

- Shape-base distances: These methods compare the overall shape of the time series [85].
- Edit-base distances: These methods count the minimum number of operations needed to transform one series into another one. These methods are also known as *Levenshtein* distance methods [88].
- Feature-base distances: These methods extracts feature that describe important aspects of time series that can be used later on any distance measures [85] [89].
- Structure-based distance: These methods find higher level structures in time series that will be used to compare time series in a global scale [90].
- Model-based distance: These methods fit a model to a set of time series and compare the parameters of the underlying model [91].
- Compression-based distance: These methods analyze how well two series can be compressed together. Then, similarity is reflected by higher compression ratio [92].

In some publications the accuracy of distance measure is evaluated within a 1-NN classifier framework. Ding et al. [85] showed regardless of different kinds of robustness, the famous DTW usually performs better in time series similarity. In the next section, we review the literature on clustering time series data sets.

3.5. Clustering

Clustering is the task of finding groups in the data set such that it maximizes the similarity between data points within a cluster while minimizes the similarity between data points in different clusters. Keogh [98] divided time series clustering into two main subtasks including *whole series clustering*, and *subsequence clustering*. Han and Kamber [99] grouped clustering methods that are developed for handling various static data into five major groups including *partitioning method, hierarchical methods, density based methods, grid-based methods,* and *model-based methods*. Brief descriptions of each categorization approach as well as their methods are described in the following.

3.5.1. Main Subtasks in Time Series Clustering

Keogh [98] considered time series clustering problem as two main subtasks. In *whole series clustering*, the clustering method is applied into each complete time series in a set. Then the entire time series set is regrouped into clusters such that it maximized the similarity of time series within a cluster and minimizes the similarity of time series in different clusters. Numerous researches have been published in the literature to explore this kind of clustering. Basically, after defining a suitable distance measure, any sort of relevant clustering method can be applied such as Self Organizing Maps (SOM) [93], Hidden Markov Models (HMM) [94], Support Vector Machines (SVM) [95].

In *subsequence clustering*, first subsequences from one or multiple clusters are extracted, and then a clustering method is applied on them to find clusters. In [96], non overlapping subsequences are extracted from time series. The lengths of the subsequences are chosen by investigating the periodical structure of the time series using DFT analysis. The limitation of this approach is when no strong periodic structure is presented in the time series. In this case, the non overlapping slicing method may miss important structures. One solution to this limitation is to extract shorter overlapping subsequences and then cluster the resulting set. However, Keogh et al. [97] showed this approach produces meaningless results. On the other hand, Patel et al. [98] showed that a meaningful clustering result can be achieved by applying a clustering mechanism on top of a motif mining algorithm.

Following this idea, Denton [99] suggested an approach that avoids using all subsequences in the clustering mechanism and produces better results. The proposed method examines the time series subsequence clustering problem from a classification perspective, where the class labels are used to identify the entire clustering performance rather than any individual clusters. The kernel density-based clustering has been used to identify potential noise subsequences that in this case the subsequence will not be assigned to the respective time series. In the proposed method, the kernel density estimator for *n* data points x_i, $i=1,\ldots,n$, in a *d* dimensional space is given by:

$$\hat{f}(x) = \frac{1}{nh^d}\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right) \tag{3.7}$$

Where the kernel function *K(x)* is normalized

$$\int_{R^d} K(x)dx = 1 \tag{3.8}$$

Gaussian kernel, equation (3.9), has been used as the most common choice for a kernel function.

$$K^{(G)}(x) = \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{|x|^2}{2}\right) \tag{3.9}$$

Then, local maxima of the density function represent the cluster centers. To identify picks of density function, all data points have been used as the starting points and through a set of hill climbing steps their location is updated. To update cluster centers the following formal is used:

$$m_h(x) = \frac{\sum_{i=1}^{n} x_i g\left(\left|\frac{x-x_i}{h}\right|^2\right)}{\sum_{i=1}^{n} g\left(\left|\frac{x-x_i}{h}\right|^2\right)} - x \tag{3.10}$$

The $g(x) = -k'(x)$ is the negative derivative of the kernel profile. The picks above a predefined threshold *t* will be considered as cluster centers. Data points are associated with the cluster center to which they are attracted. Data points that are attracted to picks below threshold *t* are considered as outliers or noises. Kernel density based clustering is robust against noise, provided the noise leads to an approximately constant density surface.

3.5.2. Major Approaches in Time Series Clustering

Most time series clustering approaches define the clustering problem as described in the following. Given a dataset of *n* unlabeled data points, a clustering method constructs *k* partitions of the dataset, where each partition represents a cluster containing at least one data point and $k \leq n$. If each data point belongs to exactly one cluster, the cluster is crisp; otherwise, it is a fuzzy clustering where one data point can be assigned to more than one cluster with different degrees of membership. *k-means* algorithm produces crisp partitions [107], where each cluster is represented by the mean value of the objects in the cluster.

Given an initial set of *k* means (centroids) $m_1^{(1)}, \ldots, m_k^{(1)}$(see below), the algorithm proceeds by alternating between two steps:

Assignment step : Assign each observation to the cluster with the closest mean

$$S_i^{(t)} = \{x_p : \left\| x_p - m_i^{(t)} \right\|^2 || \leq \left\| x_p - m_j^{(t)} \right\|^2 for\ all\ j = 1, \ldots, k\} \tag{3.11}$$

Update step : Calculate the new means to be the centroid of the observations in the cluster.

$$m_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j \tag{3.12}$$

The algorithm will converge when the assignments no longer change.

Another partitioning clustering is *k-medoids* algorithm [108], where each cluster is represented by the most centrally located object in a cluster. Both the *k-means* and *k-medoids* algorithms are partitioning the dataset up into groups, while they both attempt to minimize squared error, that is the distance between points labeled to be in a cluster and a point designated as the center of that cluster. However, *k-medoids* chooses data points as centers (*medoids* or exemplars).

Two well known methods for fuzzy partitions are the *fuzzy c-means* (FCM) algorithm [109] and the *fuzzy c-medoids* algorithm [110]. These algorithms perform very

well for spherical-shaped clusters and small to medium data sets. FCM is based on minimizing the following objective function:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^{m} \left\| x_i - c_j \right\|^2 \quad , 1 \leq m < \infty \tag{3.13}$$

where m is any real number greater than 1, u_{ij} is the degree of membership of x_i in the cluster j, x_i is the i^{th} of d-dimensional measured data, c_j is the d-dimension center of the cluster, and ||*|| is any norm expressing the similarity between any measured data and the center.

Fuzzy partitioning is carried out through an iterative optimization of the objective function shown above, with the update of membership u_{ij} and the cluster centers c_j by:

$$u_{ij} = \frac{1}{\sum_{k=1}^{C} \left(\frac{\left\| x_i - c_j \right\|}{\left\| x_i - c_k \right\|} \right)^{\frac{2}{m-1}}} \quad , \qquad c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} . x_i}{\sum_{i=1}^{N} u_{ij}^{m}} \tag{3.14}$$

The termination condition for the iteration process is $max_{ij} \left\{ \left| u_{ij}^{(k+1)} - u_{ij}^{(k)} \right| \right\} < \varepsilon$, where ε is a termination criterion between 0 and 1, whereas k are the iteration steps.

A hierarchical clustering method groups data objects into a tree of clusters. There are generally two types of hierarchical clustering methods: *agglomerative* and *divisive*. *Agglomerative* methods work in a bottom-up approach and start by placing each object in its own cluster and then merge clusters into larger and larger clusters, until all objects are in a single cluster or until certain termination conditions such as the desired number of clusters are satisfied. Whereas, *divisive* methods perform in a top-down approach and do just the opposite.

The bottleneck of a pure hierarchical clustering method is its inability to perform adjustment once a merge or split decision has been executed which results poor quality of clustering. To address this issue and improve the hierarchical clustering quality, researchers have integrated hierarchical clustering with other clustering techniques. Karypis et al. proposed Chameleon method [111] to provide careful analysis of object "linkages" at each hierarchical partitioning; Whereas, Zhang et al. developed [112] a

method named BIRCH that uses iterative relocation to refine the results obtained by hierarchical agglomeration.

Density-based clustering approaches such as DBSCAN [113], clusters are defined as areas of higher density than the remainder of the data set. DBSCAN expands a cluster until its density in the neighborhood (number of objects or data points) exceeds some threshold. DBSCAN is based on connecting points within certain distance thresholds. However, it only connects points that satisfy a density criterion, in the original variant defined as a minimum number of other objects within this radius. DBSCAN requires two parameters: ε which describes the maximum distance (radius) to consider and the minimum number of points (*MinPts*) required to form a dense region. It starts with an arbitrary starting point that has not been visited. This point's ε-neighborhood is retrieved, and if it contains sufficiently many points, a cluster is started.

Ankerst et al. proposed OPTICS [114] that rather than producing a clustering explicitly computes an augmented cluster ordering for automatic and interactive cluster analysis . The ordering contains information that is equivalent to density-based clustering obtained from a wide range of parameter settings, thus overcoming the difficulty of selecting parameter values. OPTICS also requires two parameters as ε and *MinPts*. A point p is a *core point* if at least *MinPts* points are found within its ε-neighborhood $N_\varepsilon(p)$. OPTICS also takes into accounts points that belong to a more densely packed cluster; hence, a score distance is assigned to each point that describes the distance to the $MinPts^{th}$ closest point.

$$core - dist_{\varepsilon,MinPts}(p) = \begin{cases} Undefined, & if\ |N_\varepsilon(p)| < MinPts \\ MinPts^{th}\ smallest\ distance\ to\ N_\varepsilon(p), & otherwise \end{cases} \quad (3.15)$$

The distance of another point *O* from a point *p* is called *reachability-distance* that is defined as:

$$reachanility - dist_{\varepsilon,MinPts}(o,p) = \begin{cases} Undefined, & if\ |N_\varepsilon(p)| < MinPts \\ \max\left(core - dist_{\varepsilon,MinPts}(p), dist(p,o)\right), & otherwise \end{cases} \quad (3.16)$$

Both the *core-distance* and the *reachability-distance* are undefined if no sufficiently dense cluster (w.r.t. ε) is available.

In a large multidimensional space grid-based approaches are popular where clusters are regarded as denser regions than their surroundings. Grid-based methods quantize the object space into a finite number of cells that form a grid structure on which all of the operations for clustering are performed. The computational complexity of most clustering algorithms is at least linearly proportional to the size of the data set. The great advantage of grid-based clustering is its significant reduction of the computational complexity, especially for clustering very large data sets.

The difference between grid-based clustering approaches and the conventional clustering algorithms is that it is concerned not with the data points but with the value space that surrounds the data points. In general, a typical grid-based clustering algorithm consists of the following five basic steps.

1. Creating the grid structure, i.e., partitioning the data space into a finite number of cells.
2. Calculating the cell density for each cell.
3. Sorting of the cells according to their densities.
4. Identifying cluster centers.
5. Traversal of neighbor cells.

A well-known example of the grid-based approach is STING [115], which uses several levels of rectangular cells corresponding to different levels of resolution. Statistical information regarding the attributes in each cell are pre-computed and stored. A query process usually starts at a relatively high level of the hierarchical structure. For each cell in the current layer, the confidence interval is computed reflecting the cell's relevance to the given query. Irrelevant cells are removed from further consideration. The query process continues to the next lower level for the relevant cells until the bottom layer is reached.

Model-based clustering methods analysis clusters based on probability models in which they consider a model for each of the clusters and attempt to best fit the data to the assumed model. In finite mixture models, each cluster corresponds to a component

probability distribution. The problems of determining the number of clusters and of choosing an appropriate clustering method can be recast as statistical model choice problems. Therefore, models can be compared by differences in numbers of components and/or in component distributions. Outliers are addressed by introducing one or more components that represent a different distribution for outlying data. The two major approaches of model-based methods are statistical approach and neural network approach.

An example of statistical approach is AutoClass [116], which uses Bayesian statistical analysis to estimate the number of clusters. Two prominent methods of the neural network approach to clustering are competitive learning, including ART [117] and self-organizing feature maps [118].

3.6. Classification

The classification task assigns predefined labels to each time series of a set. The goal of a classification algorithm is to first learn distinctive features that distinguishes classes from each other, and then when an unseen data is observed, it automatically determines which class it belongs to. An early approach to time series classification was first investigated by Bakshi in 1994 [100]. They used a wavelet multi scale filtering approach to extract temporal features of the time series. They combined these results with qualitative and quantitative features for a decision tree classifier. This method was applied on a chemical process trend. Later, in 1998, Keogh and Pazzani [101] proposed a piecewise representation of time series that was robust to noise. Geurst [102] used the same approach and reported that this approach is not robust to outliers.

A 1-NN classifier with DTW that is reported as the most widely used classifier [105] was shown highly accurate. However, the computing speed of this method is significantly affected by repeated DTW computations. To address this limitation, researchers in [106] proposed a template construction algorithm based on the Accurate Shape Averaging (ASA) technique. They represented each training class by only one sequence. Therefore, any incoming series is compared only with one averaged template per class.

DTW measures the distance between two time series based on their shapes. This method only uses the values of data points as the only source of information. Derivative Dynamic Time Warping (DDTW) measures the similarity of two time series based on the rate of change of the first order derivative instead of the value of the data points [110]. DDTW constructs the distance matrix of two time series of length *m* and *n*, $D = \{d\}_{m \times n}$, using the following equation:

$$d_{i,j} = (q'_i - c'_j)^2 , \quad (3.17)$$

Where q'_i and c'_j are the i^{th} and j^{th} elements in the first order derivative of two time series *Q* and *C*. The problem with DDTW is that it is only attractive to one type of shape information, because DDTW only uses the slope information of data points. The ASA method uses a Hybrid Dynamic Time Warping (HDTW) to address this problem. HDTW combines the similarity captured by both value and slope of data points. This method uses five-points stencil method to estimate the first order derivative as following:

$$f'(x) = \frac{-f(x+2)+8f(x+1)-8f(x-1)+f(x-2)}{12}, \quad (3.18)$$

Where *f(x+1), f(x+2), f(x-1),* and *f(x-2)* are the neighboring data points of *f(x)*. Since the value and the slope of the data points are of different norms and scales, the distance matrix calculation will require normalization to balance the fusion of both parameters. The normalized distance equation is proposed as:

$$d_{i,j} = \frac{d_{i,j}^{(0)} - \mu_0}{\sigma_0} + \frac{d_{i,j}^{(1)} - \mu_1}{\sigma_1}, \quad (3.19)$$

Where $d^{(0)}$ and $d^{(1)}$ are the distance computed from the values of data points and first order derivatives of the data points. The μ_0, μ_1, σ_0, and σ_1 are the means and standard deviations of the distance $d^{(0)}$ and $d^{(1)}$, respectively.

Traditional shape averaging methods computes the average between two sequences. For more than two sequences, the averaging steps are performed on the most similar pairs until one sequence is left. In each averaging operation the most similar pair is replaced by their average. Since the traditional averaging methods that use arithmetic mean cannot produce accurate results (see Figure 3-5), ASA uses HDTW-based for shape averaging to determine the optimal sequence alignment before proceeding with the averaging calculation. The shape average, *S*, is derived using the following equation.

$$S_k = \left(\frac{w_{k,1}+w_{k,2}}{2}, \frac{Q(w_{k,1})+C(w_{k,2})}{2}\right), \quad (3.20)$$

Where $w_{k,1}$ and $w_{k,2}$ are k^{th} index pair of the warping path, W.

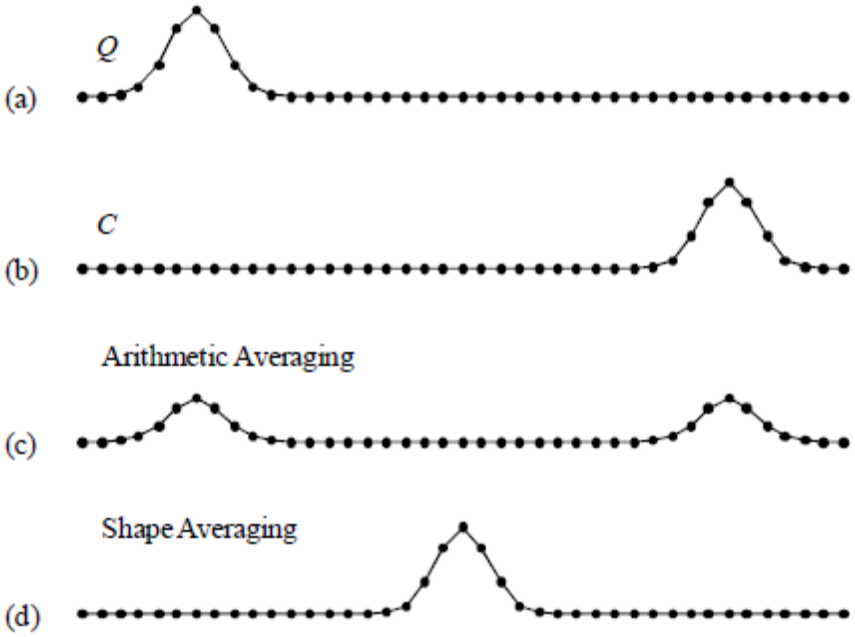


Figure 3-5. Time series averaging between sequences *Q* (a) and *C* (b) using arithmetic averaging (c) and the shape averaging (d).

To overcome the challenge of high dimensionality of time series, authors in [103] applied singular value decomposition method to select essential frequencies. But this method has a higher computational cost. Authors in [104] compared three types of classifiers: nearest neighbor, support vector machines, and decision forests. They showed that all three methods are valid, but they are highly depended on the dataset at hand.

In healthcare applications, Lin et al. [107] improved the performance of HMM using discriminative HMMs in order to maximize interclass differences. They used the probabilistic transitions between fewer states to align patients to the model and determined varying rates of progress. In a similar way, Lowitz et al. [108] applied this approach to detect post-myocardial infarct patients.

Other machine learning techniques such as neural networks [109] or Bayesian classification [110] have been studied in literature. However, Xi et al. [105] showed that many of these proposals are overpowered by a simple 1NN-DTW classifier. Subasi [111] applied a double-loop EM algorithm with a mixture of expert's network structure on

Electroencephalography (EEG) signals extracted from normal and epileptic patients to detect epileptic seizure.

Over fitting is a well-known problem in classification tasks that usually occurs because of high complexity of model and having a few number of training data points while there are too many parameters in the model. Authors in [112] applied a stopping condition to enhance the data selection during a self-training phase. Zhang et al. [113] proposed a time-series reduction method that extracts patterns to be used as inputs to classical machine learning algorithms. Many interesting applications to this problem have been investigated such as brain computer interface based on EEG signals; they have been reviewed in Lotte et al. [114].

3.7. Frequent Patterns Mining

Frequent pattern mining is the task of finding every subsequence (also known as motif) that appears recurrently in the data set. In literature, researchers sometimes call this task as "*Motif Discovery*" which is a transformation from gene analysis in bioinformatics. Patel et al. [98] defined motifs as typical non overlapping subsequences. Formally, for a given time series $T=(t_1, \ldots, t_n)$, all non overlapping subsequences $T' \in S_T^n$ that are repeatedly occurred in the original time series T are considered motifs.

Keogh et al. [97]showed that clustering subsequences by itself produces meaningless results. Instead, finding frequent patterns in time series data set can be used as a subroutine to find meaningful clusters. Chiu et al. [115] proposed an efficient method for motif discovery using random projection algorithm which produced successful results on DNA sequences. The proposed method addresses two problems in motif discovery including poor scalability of motif discovery algorithms and inability to discover motif in the presents of noise.

Figure 3-6 shows two time series with a spike noise. In plot (a), even though two time series are extremely similar, the noisy downward spike at time period 38 in one of the time series impacts the motif discovery results. Plot (b) shows the same time series (plotted in chart (a)) clustered together with a synthesis sequence, using Euclidean distance. The synthetic sequence does not particularly resemble the two real sequences, but happens to have noise in the same place as sequence 2. The dendrogram in plot (b)

demonstrates that a single piece of noise can dominate a distance function. If we allow the distance function to have "don't care" sections (denoted by the gray bar) more intuitive results can be obtained.

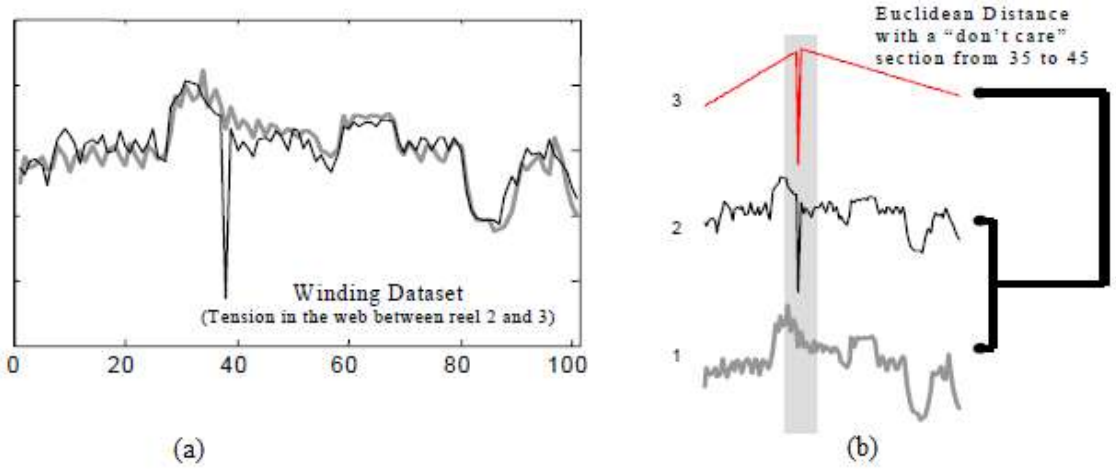


Figure 3-6. (a) The two time series from an industrial dataset. (b) The two sequences are clustered together with a synthetic sequence, using Euclidean distance. The gray bar indicates the *"don't care"* section [115].

The random projection method was introduced by Buhler and Tompa to address the planted *(w,d)*-motif problem [122], as defined below.

Planted *(w,d)*-motif problem: Given *t* strings of length *n*, that are randomly generated (i.e., each symbol generated i.i.d. with the equal probability), if each string is planted with exactly one approximate occurrence of an unknown motif *y* of length *w* (with exactly *d* substitution), find the unknown motif *y*.

The brute force search strategy that considers all possible substrings in the 2*d*-neighboorshood of all the substrings of the sequence under analysis is not practical. In fact, the size of *N(m,d)* of the *d*-neighborhood of a string *y* grows exponentially with *d* (see equation 3.15). Buhler and Tompa used random projection to reduce the huge search space by "*guessing*" at least *some* of occurrences of the unknown planted motif.

$$N(|y|,d) = \sum_{j=0}^{d} \binom{|y|}{j} (|\Sigma|-1)^j \in O(|y|^d |\Sigma|^d), \quad (3.21)$$

Buhler and Tompa projection algorithm performs in *i* iterations, where in each iteration it uniformly at random chooses *k* distinct positions among the *w* possible. The selected *k* positions are used as a "mask" to superimpose all positions on the sequences under the study. Then, by reading the symbols though the mask, each substring of size *w* is mapped to a string of size *k*. A hash table is used to store the frequency of the projected

strings. To reduce the likelihood that that some of the occurrences of the planted motif will hash together in the same entry, k is chosen such that $k<w-d$. Entries in the hash table whose count is higher than a specified threshold s are therefore selected, and they become the seed for a refinement process that uses expectation maximization. The choices of the size of the projection, k, the number of iterations, i, and the threshold, s, are crucial factors in the success of the projection method.

Although the projection method outperforms the brute force search it cannot guarantee to find the exact set of motifs. In [116] authors used protein folding/unfolding simulation to extract approximate motifs in a time series data.

Yankov et al. [117] find out that the result of a motif discovery task is highly dependent on any slight change of *uniform scaling* that is linear stretching on the length of pattern. They introduced a scaling-invariant algorithm to determine the motifs. Mohammad and Nishida [118] proposed a novel method to incorporate prior knowledge and constraints into the motif discovery process. They also proposed a method to transform most of the unconstrained motif discovery problems into constrained ones. Zhang et al. [113] applied motif discovery task on time series data set to model normal behaviors for anomaly detection which implies finding the recurrent motif of a series. They constructed motifs for each class separately to speed up the detection process.

Rakthanmanon et al. [119] proposed a new approach on clustering time series from a single stream that can be used in a motif discovery task. They described a new definition for time series clustering from streams and applied the minimum description length framework to efficiently and effectively find clusters. They showed that this method correctly finds clusters that can be used in a motif discovery task on a variety of datasets such as medicine, speech recognition, zoology, gesture recognition, and industrial process analyses.

Initializing the length of motifs is a challenging task in motif discovery. In [120], authors developed a motif visualization system based on grammar induction. The proposed method applies grammar induction to time series to effectively identify frequent patterns without prior knowledge of their lengths. Discovered motifs have variable lengths in both inter-motif subsequences and intra-motif subsequences.

Varadarajan et al. [121] introduced a novel probabilistic activity modeling approach for motif discovery on video logs. In the proposed model, each video log is considered as a document which describes a set of sequential activity patterns. This method models motifs by the co-occurrence and temporal order in which the activities (words of documents) occur within a temporal window. This model allows overlap on the advent of activity motifs. They showed promising results on synthetic data and four video datasets with significant variations in their activity content.

3.8. Conclusion

In this chapter we reviewed literature related to time series mining. Clustering, classification, indexing, and motif discovery are main challenges in time series data mining that we covered in this chapter. Following chapters present our approaches in defining a new similarity measure for time series, classification, clustering, and motif discovery on time series dataset.

Chapter 4 Temporal Smith Waterman Algorithm

Rapid aging of the population in the US requires increased attention from health care providers and from the entire society as a whole. A possible solution to prevent unreported health problems in independently living older adults is through automatic health monitoring systems. One efficient approach to health monitoring is to use sensor networks to collect information about the older adult's activity. As we mentioned in chapter 1, in TigerPlace, our "living laboratory", we have installed sensor networks in the apartments of our residents to consider the health context of the monitored older adults. In chapter 2, we described a version of our framework system for early illness recognition. This framework uses Euclidean distance and DTW to measure the similarity of two time series. While Euclidean distance performs weak in the presence of noise, DTW solves this challenge; however, its performance in multi-dimensional sensor time series is not desirable. In this chapter, first we review related works to measure the similarity of two multi-dimensional time series. We continue with describing our novel approach, a modified version of SW algorithm, TSW.

4.1. Related Works

Sensor networks have been used in last decade as a promising solution to monitor the health of older adults. One possible approach for the automatic recognition of health problems is to assume that similar medical conditions result in similar behavior, hence in similar sensors activity patterns. While this assumption may not be generally true for a younger, more mobile population, it has a certain degree of validity for the elderly [127] [128] [129]. To better understand an older adult's behavior, elderly monitoring systems may use multiple sensors, such as motion, Kinect, radar, sound, etc. [9]. Typically, in eldercare applications the sensors are not wearable, but rather placed in the living environment to collect data about the residents' behavior.

During the analysis, the task of assessing behavior similarity can be cast as computing the similarity of two MATS. MATS similarity is a challenging task and is still an open research question. Here we mention two most popular approaches to this task.

First approach consists of using the Euclidian distance as a measure of similarity between two sequences [130]. Euclidian distance is sensitive to outliers, cannot capture time and cannot be computed for sequences of variable length. The main limitation of using the Euclidean distance as a similarity measure is due to its weak performance in the presence of noise. The second commonly used approach to MATS similarity is based on the non-Euclidean metrics, such as DTW [27]. Authors rephrased the problem of finding similar time series instead as a problem of sequence alignment. They presented an algorithm for DTW on multi-attribute time series and compared it to the ordinary DTW, where a single attribute is used for sequence alignment. The DTW algorithm for two sequences calculates the distance between each possible pair of data points. Then, the algorithm uses these distances to calculate a cumulative distance matrix, and it finds the best path through this matrix.

The second distance measure in the family of non-Euclidean measures is a modified LCSS for continuous domain [131]. LCSS offers more robustness in the presence of noise compare to DTW. Authors in [131] showed that the computational time required by this algorithm is quadratic to the sequence (in our case, the time series) length. To address this limitation, they applied upper boundary functions to limit the computational time. They utilized a fast pre-filtering schema that returns upper bound estimations for the LCSS similarity. In a similar way, authors in [132] proposed a Multi-Dimensional Dynamic Time Warping technique, which modifies the distance matrix by using the vector norm to calculate the distance between a pair of points. Although both measures presented a significant improvement compared to the Euclidean distance, some of their drawbacks are: quadratic time complexity, computational overload of the boundary detections, and filtering of data, which may remove valuable information.

In applications where the number of training samples is sufficient, Hidden Markov Models typically outperform DTW. In [133], authors compared ordinary DTW, a probability-based DTW and a HMM for the sequence comparison. Using a Gaussian-based probabilistic model for a gesture recognition application, the probability-based DTW shows better results. In [133] a soft distance, based on posterior probability of the Gaussian Mixture Model, was defined. Despite of HMM's performance, there are several

problems with this approach. They include the need for a large amount of data to train HMM, making strict assumptions about data, and setting a large number of parameters.

Our novel approach to sequence similarity is to consider MATS as one-dimensional sequence and to use bioinformatics techniques to find the best alignment. This approach is possible when sensor data is already quantized by the hardware system, which results in a symbolic sequence of sensor firings and the related firing time. SW algorithm is a well-known algorithm to find an optimal local alignment [14]. SW algorithm calculates a local similarity score in a linear space, if we just look for a local alignment. In bioinformatics, one motivation for the local alignment is in a situation where biological sequences are highly related to each other, but the similarities in local regions are very low. In this case, it is hard to find the correct alignment. To tackle this problem, a local alignment discards such regions and concentrates on those with high similarity. Another motivation for local alignment is to use a reliable statistical model for optimal local alignment to produce expectation values [30].

4.2. Temporal Smith Waterman

To predict early illness from combining the non-wearable sensor data with medical concepts extracted from nursing notes, we employ a NLP methodology. There are several challenges that we intend to address. First challenge is to identify the most suitable similarity measure for comparing multi-dimensional time series. Second, we want to improve the illness prediction using sensor series similarity and utilizing sets of UMLS concepts. Finally, we employ aggregation methods to infer the most probable health concepts associated to an unknown sensor sequence.

Table 4-1 shows the pilot sensor data from apartments of three TigerPlace residents in this study. For each resident, we also retrieved visit notes about physical, emotional, and other health complaints, recorded by nurses at TigerPlace which are stored in EHR system [135]. In our dataset, there are fewer notes than sensor data (automatically logged for each day per resident), as some days didn't have any nursing comments. However, some residents (for example, #3) have days with multiple comments. In addition, each day was manually annotated based on the EHR data (nursing comments, blood pressure, etc.) as normal or abnormal. The abnormal days (column 4 in

Table 4-1) were subjectively determined by retrospectively inspecting the nursing notes. We only use them to tune our sequence similarity algorithm.

Table 4-1.TigerPlace pilot dataset for TSW experiments.

Resident Code	Number of sensor days	Number of comments	Number of abnormal days
#1	440	83	81
#2	745	44	35
#3	500	499	335

While we acknowledge that our dataset is small, we mention that it has to be seen as three separate experiments with more than 400 samples each, rather than one experiment with only three samples. The reason for this perspective is that mathematical models (such as classifiers, algorithm parameters) for early illness recognition are not usually transferrable from one patient to another, because the disease-behavior associations could be vastly different between people. In addition, in this work we merely intend to open a research direction rather than claiming that our framework is ready for clinical trials.

4.2.1. TSW Algorithm

Sensor networks have previously been used in health care to predict health patterns. Defining a suitable distance/similarity measure to find similar patterns in sensor networks data is still an open question. The multi-dimensional nature of the sensor network complicates the task of measuring similarity. In chapter 2, we used aggregated sensor data time-wise (in one hour time intervals) and type-wise (aggregated all motion sensor data together) obtaining a three-dimensional sequence vector. We used a RMS approach, a widely used techniques in signal processing to find the distance between time series. That is, for two sensor sequences $X = \{x_{ij}\}$, $Y = \{y_{ij}\} \in R^3 \times R^{24}$ we can compute the RMS distance as:

$$RMS(X,Y) = \sqrt{\frac{1}{24}\left((X_1 - Y_1)^2 + \cdots + (X_{24} - Y_{24})^2\right)} \quad ,$$

where $X_i = \sqrt{\frac{1}{3}\left(x_{1i}^2 + x_{2i}^2 + x_{3i}^2\right)}$ $\qquad$ $Y_i = \sqrt{\frac{1}{3}\left(y_{1i}^2 + y_{2i}^2 + y_{3i}^2\right)}$.

In this work, we propose a new approach where we don't aggregate the data and we maintain the order of the sensor firings as recorded in the log file. To find similar

sensor firing sequences, we employ SW algorithm. However, the drawback of SW algorithm is that it does not consider time in local alignment. Time is an important factor in sensor firing sequence, because various time intervals can be associated with different behaviors (for example fast or slow walking). In this paper, we propose a methodology based on SW algorithm that accounts for time in sequence similarity.

SW algorithm was first proposed in [14] to align two molecular sequences, $T_1=\{C_{11}, C_{12}, \ldots, C_{1m}\}$ and $T_2=\{C_{21}, C_{22}, \ldots, C_{2n}\}$where $m,n \in \mathbb{N}$ and C_{ij} belongs to alphabet Σ (nucleotides or amino acids). First, SW builds a similarity score matrix H where $H(i,j)$ is equal to the similarity score between a suffix of $T_1[1\ldots i]$ and a suffix of $T_2[1\ldots j]$ such that

$$H_{i0}=H_{0j},\ i\epsilon[1,n] \text{ and } j\epsilon[1,m] \quad (4.1)$$

$$H_{ij}=max\{0, H_{i-1,j-1}+S(C_{1i}, C_{2j}), max_{k\geq 1}\{H_{i-k,j}-W_k\}, max_{k\geq 1}\{H_{i,j-k}-W_k\}\} \quad (4.2)$$

$$W_k = g + ck \quad (4.3)$$

$$FinalScore = \frac{max\{H_{ij}\}}{min\{n,m\}} \quad (4.4)$$

Figure 4-1. Smith Waterman (SW) algorithm.

In equation 4.2, $S(C_{1i}, C_{2j})$ is the similarity between symbols $C_{1i}, C_{2j}\epsilon\Sigma$ denoted henceforth as S_{ij} . The penalty of opening a gap will be calculated by equation 4.3 where g and c are two constants and $k\epsilon\mathbb{N}$ is the length of the gap. The final alignment score is derived from equation 4.4. Based on these calculations, for each character-to-character comparison SW calculates a score that is positive for exact matches or substitution, and negative for insertion or deletions. In weight matrices, scores are added together and the highest scoring alignment is reported.

To consider time as a factor in calculating the similarity score, we propose a temporal variant of the SW algorithm (TSW) by considering time as a gap. In order for this approach to work, we assume that each character in the sequence has an associated time stamp. The time stamp shows the time when a character (firing) is emitted by the related sensor network. Consider two time-stamped sequences of characters as $T_1=\{(C_{11}\ t_{11}), (C_{12}\ t_{12}), \ldots, (C_{1m}\ t_{1m})\}$ and $T_2=\{(C_{21}\ t_{21}), (C_{22}\ t_{22}), \ldots, (C_{2n}\ t_{2n})\}$ where $m,n \in \mathbb{N}$ and C_{ij} belongs to alphabet Σ and t_{ij} are the time of the day of the firing C_{ij}. Figure 4-2 shows the temporal SW algorithm.

$$H_{i0}=H_{0j},\ i\epsilon[1,n]\ and\ j\epsilon[1,m] \quad (4.5)$$

$$H_{ij}=max\{0,\ H_{i-1,j-1}+S(C_{1i},\ C_{2j}),\ max_{k\geq 1}\{\ H_{i-k,j}\ -\ W_{\Delta t}),\ max_{k\geq 1}\{\ H_{i,\ j-k}\ -W_{\Delta t}\}\} \quad (4.6)$$

$$W_{\Delta t}\ =\ g\ +\ c|t_{1i}-t_{2j}| \quad (4.7)$$

$$FinalScore\ =\ \frac{max\ \{H_{ij}\}}{min\ \{n,m\}} \quad (4.8)$$

Figure 4-2. Temporal Smith Waterman (TSW) algorithm.

Essentially, TSW considers the time of the day between the sensor firings as a gap and computes the gap penalty $W_{\Delta t}$ by using time stamps (as shown in equation 4.7 above). We use the "time of the day" metric since we would like to find similar behaviors across different days that happen at roughly the same time. Note that the type of time used in (4.7) is critically important and is dependent on the application. For example, if one would like to compare patients represented by the sequences of ICD-9 diagnoses, the time would more likely include the year of each diagnosis. An interesting aspect of our algorithm is one doesn't need to know in advance the time unit required by equation (4.7). Instead, the search for best constant *c* should be able to discover the scale of the events as we will see in the experimental results section.

4.2.2. Sensor Sequence Similarity Evaluation

To evaluate our sensor sequence similarity measure, we used the data from Table 4-1, column 2 and 4 (sequences with normal/abnormal labels). Then, we employed a *k*-nearest-neighbor (*k*-NN) method with leave-one-out approach to classify each sequence as normal or abnormal. To test our TSW similarity measure, we found the *k*-nearest sequences (*k*=5) to test one and get the majority vote for the class label (normal or abnormal day). The results are reported in terms of *Sensitivity* and *Specificity*, that is:

$$Specificity\ =\ \frac{number\ of\ true\ negatives}{number\ of\ true\ negatives+number\ of\ false\ positives} \quad (4.9)$$

$$Sensitivity\ =\ \frac{number\ of\ true\ positives}{number\ of\ true\ positives+number\ of\ false\ negatives} \quad (4.10)$$

Finally, we compared the classification results obtained using TSW to the ones provided by RMS similarity presented in chapter 2.

4.2.3. Sensor Sequence Annotation Using TSW Algorithm

Computing sensor sequence similarity allows us to proceed toward our goal of predicting patient's health status using sensor data and EHR medical concepts. We obtained medical concepts by parsing the text data from the nursing notes using the Metamap NLP tool provided by UMLS (http://metamap.nlm.nih.gov/). Metamap associates each medical concept found in the nursing notes with a CUI. The prediction algorithm used a KNN method with a leave-one-out cross-validation approach. For each unknown sensor sequence X_i, we compute the distances (using both TSW and RMS for comparison) to all other daily sequences. Then we select k ($k=5$) most similar sequences and their associated CUIs, T_j, j=1,k. Finally, we assign to X_i the medical terms with CUI's that are computed as the intersection of all T_j that is we take the CUIs common to all retrieved sequences. The annotation algorithm is given in Figure 4-3below:

Illness recognition algorithm using Temporal Smith Waterman

1. Initialize k, g, c;
    a. k = number of neighbors
    b. g = gap opening
    c. c = gap extend
2. For each sensor sequence X_i, compute its distance (TSW or RMS) to all N sequences
3. Find k most similar sequences X_j, j=1,k
4. Extract UMLS terms T_j (X_j)= { CUI's associated with X_j }, j=1,k
5. Annotate sequence X_i, with the UMLS terms $T_i = \bigcap_{j=1}^{k} T_j$

Figure 4-3.Illness recognition algorithm.

We used *precision*, *recall*, and *F-measures* for the evaluation of the algorithm, defined as:

$$precision = \frac{True\_Positive}{True\_Positive + False\_Positive} \quad (4.11)$$

$$recall = \frac{True\_Positive}{True\_Positive + False\_Negative} \quad (4.12)$$

$$F\_measure = 2\frac{precision * recall}{precision + recall} \quad (4.13)$$

During these experiments, we only used the days from our data set that have associated nursing notes (see column 3 in Table 4-1). The results are given in the next section.

4.3. Experimental Results

To investigate the performance of the proposed illness prediction methodology based on TSW, we compare these results to the results that described in chapter 2 which used multi-dimensional sensor time series and RMS distance. Our evaluation has two steps. In the first step, we investigate the properties of the TSW similarity and compare it to RMS. In the second step, we demonstrate the results of illness prediction based on TSW sensor sequence similarity.

4.3.1. Results for TSW Similarity Measure

Before comparing TSW to RMS we have to choose the best values for the temporal gap parameters g and c (see formula 4.7). For this purpose, we compute the sensitivity of classifying each sequence as normal/abnormal using the dataset shown in Table 4-1 (columns 2 and 4) and the method described in section 4.2.1. To limit our search space we set $g=0$. The variation of the Sensitivity with c for resident #1 is shown Figure 4-4. Note that times t_{ij} in formula 4.7 are given in seconds.

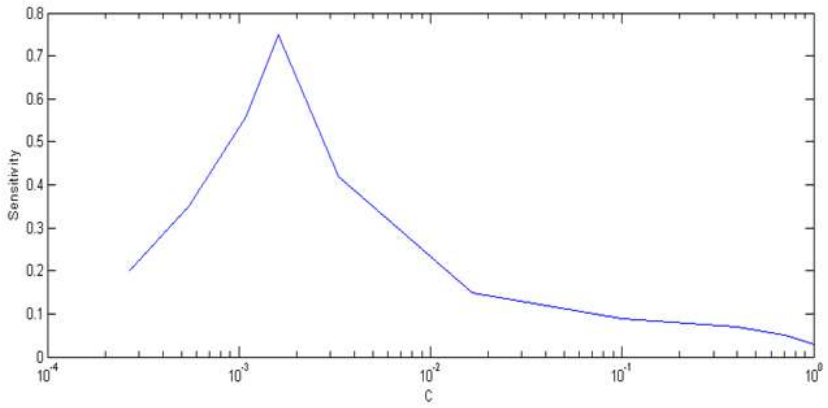


Figure 4-4. Variation of sequence classification sensitivity vs. temporal gap parameter c.

We see that the sensitivity has a sharp maximum around c=0.002. Since the times in (equation 4.7) are in seconds, it implies that our time scale is effectively about $1/c$~480s/60~8 min. We obtained similar results for the other two residents.

The classification results obtained using TSW and RMS on the sensor sequences mentioned in Table 4-1 (column 2 and 4) as normal/abnormal are shown in Table 4-2 and 4-3, respectively.

Table 44-2. Sensor sequence classification using TSW similarity

Resident	***Sensitivity***	***Specificity***
#1	0.89	0.36
#2	0.96	0.53
#3	0.48	0.85

Table 4-3. Sensor sequence classification using RMS similarity.

Resident	***Sensitivity***	***Specificity***
#1	0.66	0.27
#2	0.71	0.10
#3	0.16	0.67

As it can be seen from above tables, TSW clearly outperforms RMS by about 20-30% in sensitivity and 10-40% in specificity. Obviously, sensor aggregation (both temporal and by sensor type) is detrimental to the sequence classification. Interestingly, the sensitivity results for resident #3 is lower than for the other two elders, for both TSW and RMS algorithms. The reason for the lower sensitivity is that the classification of abnormal days is not as reliable as for the normal days. The resident #3 has the greatest proportion of abnormal days in the dataset (has more medical problems than the other two). This could be caused by both the diversity of abnormal days (multiple behaviors, possible very different) but also by the subjective labeling of days as abnormal.

4.3.2. Results for Illness Prediction

In the second set of experiments, we performed illness prediction by annotating sensor sequences with UMLS concepts from the nursing notes. We describe this process by an example shown in the Figure 4-5.

In Figure 4-5 we have a sensor sequence of resident #2 on February 15, 2006, and we want to classify this day as "normal" – no annotation available or "abnormal"- find all related UMLS terms. The system processes this sensor sequence with the TSW algorithm to detect the most similar sensor sequences to this pattern in the stored database of all past motion sensor sequences for this resident. Then, the system extracts nursing notes associated with the selected similar days detected by the TSW algorithm. We apply NLP and UMLS to convert raw text (nursing notes) to a set of CUI's (concepts unique identifiers of the medical terms). Finally, the system identifies the sensor sequence of 2006-02-15 as an abnormal day because similar days, such as 2006-02-13 and 2006-02-11, are tagged as "abnormal", and the system also proposes a set of CUI's as suggested abnormalities concerns ("pain" and "antidepressant").

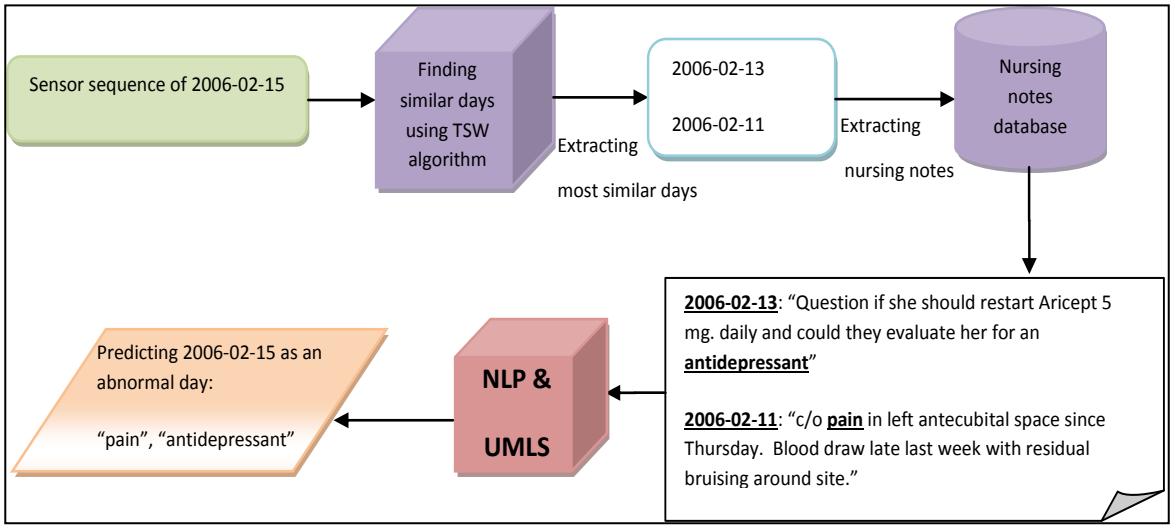


Figure 4-5. Predicting abnormal days using nursing notes.

While the final version of the system uses all available sensor history, in the following experiments we only used the abnormal days (column 4 in Table 4-1), that is the search was conducted only among the "abnormal" days. The scaled down experiments were necessary to increase speed as our TSW implementation is not time-efficient at this time. Moreover, including the "normal" days requires solving

supplementary problems such as what to do when out of three most similar sequences two are “normal” and one “abnormal”. The comparison of the annotation results obtained with the TSW and the RMS measures are shown in Tables 4-4 and 4-5, respectively.

Table 4-4. Sequence annotation using TSW similarity.

Resident	***precision***	***recall***	***F_measure***
#1	0.62	0.37	0.46
#2	0.41	0.9	0.55
#3	0.44	0.57	0.47

Table 4-5. Sequence annotation using RMS similarity.

Resident	***precision***	***recall***	***F_measure***
#1	0.51	0.52	0.42
#2	0.41	0.34	0.35
#3	0.35	0.54	0.35

As it can be seen from above tables, the *F-measure* improved by 4-12% by using the TSW measure, but the improvement is not as significant as the one observed in Tables 4-2, 4-3. The main reason of this outcome is that in this experiment we did not use the “normal” days, that is, the days without any annotation. This could be easily done by labeling each day without any nursing note with the term “normal” and perform the same experiments as before.

4.3.3. Study Limitations

We acknowledge that our study has the following limitations:

1. Our study is based on the fundamental assumption that there is a univocal relation between disease states and behavior. While we acknowledge that, for younger,

active adults this assumption does not hold, we believe that this becomes increasingly true for older, less active adults.

2. Our patient dataset is small. However, it is not insignificant: it contained 1685 days of sensor data (85.8 million sensor firings) and 451 nursing notes. We have 47 sensor networks deployed at TigerPlace and we plan to extend our study to include our entire resident population.

3. We did not include all of the available sensors such as Kinect, radar and microphone array. Including vision based sensors, such as Kinect, can greatly increase behavior recognition capabilities. We plan to include Kinect data in the future studies.

4. Our sensor sequences represented one 24-hour day, from 0h to 23:59h, which would not allow us to capture the similarity of two days, say $\{d_1, d_2\}$, where d_1 had one event 5 minute after midnight (hence at the beginning of the sequence) and d_2 had the same event 5 minutes before midnight (hence at the end of the sequence). In the next chapter, we implemented the modification of TSW to search for the similar sequence using a sliding window over the entire time frame.

4.4. Conclusions

In this chapter, we presented a framework for illness prediction using sensor networks and a temporal Smith Waterman (TSW) algorithm. The health patterns were described by NLM, CUI terms that were extracted from nursing notes using Metamap, a NLP tool released by NLM. The associations between nursing notes and sensor sequences were made possible by our unique nursing notes stored in EHR that captures both sources of information for the TigerPlace residents. We tested our framework on a pilot set that consists of 1685 days of sensor data and 451 nursing notes. We compared our results with an approach that uses sensor aggregation and a Euclidean-type (RMS) similarity measure. In this comparison, the proposed TSW framework outperformed the one based on RMS. In the next chapter, we describe the modified framework that has more sensors and residents to our study and refines the annotation procedure.

Chapter 5 Analysis on Temporal Smith-Waterman Algorithm

In the last decade, data mining techniques have been applied to sensor data in a wide range of application domains, such as healthcare monitoring systems, manufacturing processes, intrusion detection, database management and others. Many data mining techniques are based on computing the similarity between two sensor data patterns. A variety of representations and similarity measures for multi-attribute time series have been proposed in literature. In chapters 3 and 4, we reviewed some of them and proposed a novel similarity function to compare two sensor sequences. In this chapter, we address study limitations of the approach described in the previous chapter. We analyze a novel method for computing the similarity of two multi-attribute sensor time series based on a temporal version of Smith-Waterman, a well-known bioinformatics algorithm. We then apply our method to sensor data from an eldercare application for early illness detection. Our method overcomes difficulties related to data uncertainty and aggregation that often arise when processing sensor data. To validate our method we used data from nine non-wearable sensor networks placed in TigerPlace apartments, combined with information from an EHR. We provide a set of experiments that investigate TSW properties, together with experiments on TigerPlace datasets. On a pilot sensor dataset from nine residents, with a total of 1462 days of collected data, we obtained an average illness prediction accuracy of 70%.

5.1. Introduction

Wireless based health monitoring system is a successful solution to address challenges, such as comprehensive wireless health monitoring system, context awareness, reliability, and autonomous adaptability of a wireless health monitoring system. Elite Care [136] uses sensors to monitor health status of residents. The system offers vital signs, health alarms and indicators to personalize environment for residents. The ad hoc wireless network, LANs, cellular/GSM/3G networks and satellite-based systems have been used to set up a pervasive healthcare solution for heath monitoring, emergency management, intelligent health care data access, and mobile telemedicine [137] [138].

The healthcare monitoring system assesses the health status of elder residents via the physiological parameters such as ECG, heart rate, body temperature, as well as motion patterns, and visual detection of elderly patient. The system detects potentially abnormal physiological parameter values, sends medical alerts, and predicts activity patterns, such as rapid movement or lack of movement of elder resident.

A key feature of these health-monitoring systems is the ability to continuously and unobtrusively collect information about daily activities of older adults. These systems process the acquired sensor data, infer the activities and behaviors of the older individuals, and detect changes in their health status. Our goal is to automatically detect changes in health status using sensor data provided by a non-wearable sensor network. Early sign of an impending illness or an exacerbation of an existing chronic condition may produce detectable behavior changes. The link between behavior changes and health patterns is based on the assumption that similar medical conditions result in similar (abnormal) behaviors, hence in similar sensor patterns. If a sensor pattern is not similar to previous ones observed in similar context (time, location), then we could assume that is produced by some (unknown) health conditions. While these assumptions might not in general be true for a younger, more mobile population, they have a certain degree of validity for the elderly [127] [128] [129]. To understand a person's behavior, researches employed various sensors, such as motion, Kinect, radar, sound, etc. [9] that produced MATS. Consequently, the task of assessing behavior based on sensor data can be implemented based on computing the similarity of two MATS.

There are many approaches to computing MATS similarity depending on the application and the attribute type (discrete or continuous). In chapter 2, we discussed Euclidian and DTW distance to compute the dissimilarity of two sequences of continuous MATS of equal length. However, the Euclidian distance is sensitive to outliers and cannot be used for sequences of different lengths. To overcome these challenges, one can use algorithms based on DTW for length mismatch or LCSS in the presents of outliers [27] [131] [132]. The DTW algorithm for two sequences calculates the distance between each possible pair of their points. Given these distances, a cumulative distance matrix is calculated and the best path through this matrix is found. LCSS similarity is reportedly more robust to noise than DTW [131]. The main difficulty of LCSS and DTW which are

based on dynamic programming are their time requirements. Faster, alignment free algorithms based on subsequence statistics are being developed in bioinformatics for computing the similarity of genomic sequences [139].

While in bioinformatics sequences are mostly discrete (symbolic), they tend to be mostly continuous (numeric) in financial applications and mixed (some dimensions are numeric- heart rate, and others are symbolic- motion detectors) in eldercare applications. While some similarity measures such as LCSS [27] or SW [14] are originally designed for discrete data they can be converted to continuous data by either modifying the original algorithm [27] or converting the sequence from discrete to continuous format [128] [140] [141]. The disadvantage of this approach is that it results in information lose.

In this chapter, we describe a similarity approach for discrete time series and its applications to eldercare. Our approach doesn't require series data conversion to continuous format. Instead, we arrange all different sensor hits together with their time stamps into a one-dimensional sequence. In chapter 4, we introduced the idea of modified SW [14]framework, TSW, for early illness recognition that uses time-stamped sequences. We represented the sensor time series of each 24-hour day as a one-dimensional sequence and compared sequences for two different days. However, the question is whether this generalizes to comparing sensor sequences that are only several hours long (e.g. 5 hours)? In this chapter, we analyze the properties of the TSW, introduce a faster way to compute it based on genetic programming and propose a new method for searching large time series. In the end, we show how TSW similarity can be used for detecting abnormal health patterns using non-wearable sensor networks.

Following, in section 5.2 we describe our system architecture and data sets. In section 5.3, we explain our methods. In section 5.4, we provide extensive experiments to investigate our methods. Finally in section 5.5, we discuss the implications of this approach.

5.2. System Architecture

In this chapter, we used the same structure as described in the previous chapter. Here we extended our experiments by adding more residents to our dataset. Table 5-1

describes the pilot sensor data from nine residents of TigerPlace used in this chapter with a total of 1902 sensor-days. For each resident, we also retrieved visit notes about various physical and/or mental health complaints added by the nurses on site to the resident's EHR. In our dataset there are fewer notes than sensor data (automatically logged for each), as for some days there were no nursing comments. However, some residents (e.g. resident #3) may have multiple comments per day. In addition, each day was manually coded as normal or abnormal for each individual resident. Clinical experts retrospectively reviewed the TigerPlace EHR data (nursing comments, vital signs, etc.) to flag days (denoted as "abnormal") with resident's illness episodes (column 3 in Table 5-1). We use the abnormal days to tune our sequence similarity and test our early illness recognition algorithm.

Table 55-1. Pilot Dataset for modified TSW experiments.

Resident Code	**Number of sensor days**	**Abnormal days**
1	440	83
2	463	44
3	499	335
4	225	130
5	81	60
6	72	55
7	45	15
8	39	10
9	38	15

5.3. Analysis on TSW Algorithm

In this section, we describe TSW similarity measure, a search procedure based on TSW and a method to speed it up.

5.3.1. Temporal Smith-Waterman Similarity

We define a time series S as a set of n couples $(s_i\ t_i)$, i.e. $S=\{(s_1\ t_1),\ \ldots,\ (s_n\ t_n)\}$. Each couple $(s_i\ t_i)$, $1 \le i \le n$, has two components: a sensor signal s_i that belongs to a symbols set Σ and a time stamp t_i which represents the time when sensor s_i was recorded. The alphabet Σ is a set of identifiers we use to represent MAST. In our case, Σ comprises of the symbols shown in Table 2-1.

While some sensors such as motion detectors can be naturally described by symbolic data, others, such as bed motion sensor may need to be quantized. That was the case of our bed motion sensor for which the bed motion was empirically divided [12] in 4 categories: less than 3 s, 3-7 s, 7-14 s, more than 14 s (sensor ID 1-4 in Table 2-1).

Given two discrete time series $S_1=\{(s_{11}\ t_{11}),\ldots,(s_{1n}\ t_{1n})\}$ of length n and $S_2=\{(s_{21}\ t_{21}),\ldots,(s_{2m}\ t_{2m})\}$ of length m, with $s_{1i},s_{2j}\in\Sigma$, $1 \le i \le n$, $1 \le j \le m$, we can compute their TSW similarity, TSW(S_1, S_2), using the following algorithm:

$$H_{i0}=H_{0j}, \qquad i\in[1,n] \text{ and } j\in[1,m] \tag{5.1}$$

$$H_{ij}=max\{0,\ H_{i-1,j-1}+Sim(s_{1i},\ s_{2j}),\ max_{k\ge1}\{H_{i-k,j}-W_{\Delta t}),\ max_{k\ge1}\{\ H_{i,\ j-k}-W_{\Delta t}\}\} \tag{5.2}$$

$$W_{\Delta t} = g+c|t_{1i}-t_{2j}| \tag{5.3}$$

$$\text{TSW}(S_1,S_2)=max\{H_{ij}\}/min\{n,m\}, \tag{5.4}$$

where H is a work matrix used in the dynamic programming to trace back the best alignment between S_1 and S_2 and "*Sim"* is a symbol similarity matrix that reflects the compatibility between symbols. For example, *Sim*(BedMovement1, BedMovement2)=0.9 since they are both bed motion sensor firings (see Table 2-1), whereas *Sim*(BedMovement1, Cabinet)=0 since they belong to different types of sensors. The g constant is a penalty for opening a gap, while c is a penalty for extending a gap. We take $g=0$ in this paper.

Similar to the traditional SW used in bioinformatics [14], TSW considers the time difference between two sensor firings as a gap and, consequently, computes a gap penalty $W_{\Delta t}$ using the time stamp associated to each symbol (Equation 5.3). The time is given in seconds and the constant c used in equation 5.3 controls the time scale associated with the symbol firing. While the TSW algorithm can be used for any symbolic time series, the time scale used in equation 5.3 is application dependent. For example, in computing the similarity between patients represented by sequences of International Classification of Disease version 9 (ICD-9) diagnoses [29], the time scale would probably be of the order of months where in our application the time scale is of the order of minutes. The exact computation of c can be performed if a training set of known sequence similarities is available, as we will show in the result section.

5.3.2. Sequence Search Using TSW

Our main motivation for developing a sensor sequence similarity measure is to be able to compare human behaviors and activities. One simple approach that we described in chapter 4 is to divide the entire sensor sequence acquired from a TigerPlace apartment into fixed intervals such as days (sets of 24-hour long sensor sequences) and use the TSW to compute the similarity between them. However, since a given activity can be performed at different times of the day, week or year, we should be able to search the entire database for a given pattern. To address this question, in this chapter, we employ a Window based TSW (WTSW) method.

WTSW algorithm uses a sliding window approach to find the similar subsequences to a user defined (query) subsequence based on TSW similarity measure. In our case, examples of user defined subsequences are bathroom visits or food preparation activities. Suppose we denote the sequence of past sensor firings by D and the user defined subsequence (query) by Q. To find the most similar subsequence to Q in D, we slide a window of size t_{Δ} over D. The process is shown in Figure 5-1. The size of the window, t_{Δ}, must be greater than the difference of the times when S_{q1} and S_{qn} were observed in Q, that is $t_{\Delta} > (t_{qn}-t_{q1})$. Consecutive windows overlap by an interval of time $(t_{\Delta}-W_{\Delta})$. To exclude trivial matches (subsequences part of Q), we don't consider subsequences with time stamps that overlap with Q. After extracting all of non-trivial

subsequences, we assign a similarity score to all of subsequences using TSW similarity measure. We then pick the subsequence with the highest score as the most similar subsequence to the given user defined subsequence. The pseudo code of the WTSW algorithm is represented in Figure 5-2.

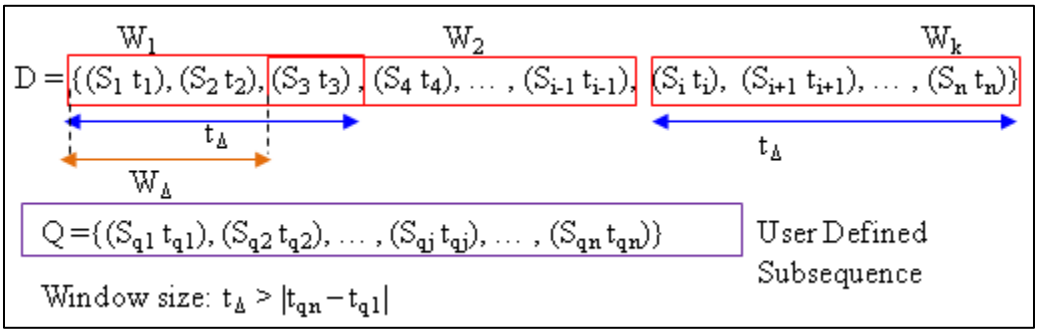


Figure 5-1. Finding the most similar subsequence using WTSW method.

Step 1: Initialization

Initialize parameter W_Δ

Calculate parameter t_Δ using user defined subsequence Q;

Form a window of size bigger than t_Δ from the start point of data set sequence;

Initialize subsequence set $W = \Phi$;

Step 2: while (termination condition)

{

Form W_i = the subsequence (i) of size bigger than t_Δ

Calculate the similarity score of W_i using TSW algorithm

$Score(W_i) = TSW(W_i, Q);$

Add subsequence W_i to the set W;

Slide the window at the rate of W_Δ to get the next subsequence;

}

Step 3:

Pick the highest score W_i for the set W;

Figure 5-2. Pseudo code of WTSW method.

The WTSW faces two challenges. The first challenge is speed. The TSW algorithm is fairly time intensive. The second challenge is initializing parameter W_Δ. If W_Δ is small, the results are more accurate but the computing time is high. The running time can be decreased by reducing the overlap, but some relevant subsequences might not be retrieved. The TSW similarity of two subsequences of lengths m and n, has $O(mn)$

complexity. If length of D is r, then we might need between r/n (no overlap) and r-n (n-1 overlap) TSW evaluations. If $r>>n$, then the upper bound of complexity is rn^2, which may be high for large r. For example, a "bathroom visit" behavior has typically 200 symbols (n=200) and the 5 years past behavior sequence may be 2 million symbol long (r=2,000,000).

To address these two challenges, we employ the genetic algorithm approach described in the following section.

5.3.3. Speeding of WTSW using a Genetic Algorithm

In this section, we propose optimized WTSW using genetic algorithm, and we call it as Genetic Algorithm-based Temporal Smith Waterman (GATSW) method. Using a genetic algorithm to obtain an optimal solution for the WTSW requires that we define the relevant parameters and their encodings in terms of chromosomes, genes, and populations.

Definition 5.1. Gene: Given a subsequence S of length m, where S= {(s_1 t_1), …, (s_m t_m)}, a gene is a pair (s_i, t_i) , $1 \le i \le m$, that identifies the sensor S_i that fired at time t_i. That is, a gene in WTSW is a window representing a sensor firing and its time stamp.

Definition 5.2. Chromosome: A chromosome is a sequence of sensor firings within a specified time interval, namely a set of genes. Given a time series T of length n, a subsequence S_i of T is an ordered continue sampling of length m, S_i = {(s_i t_i), …, (s_{i+m-1} t_{i+m-1})} where $m \le n$, $1 \le i \le n - m + 1$.

Definition 5.3. Population: A population of size p is a set of p different chromosomes.

In a genetic algorithm, we use a fitness function to determine which chromosomes survive from one generation (iteration) to another.

Definition 5.4. Fitness function: In GATSW, the fitness of a chromosome S_i is calculated by assessing the similarity of the chromosome to the given user defined subsequence Q. The fitness of chromosome S_i is

Fitness(S_i)= TSW(S_i, Q).

Another fundamental operation to derive a new generation is mutation. Mutation is the process of generating children chromosomes from the parent chromosome(s). Here we define mutations that mimic the process of sliding a window.

Definition 5.5. Mutation: Given a chromosome S_j, a mutation over S_j changes the start position based upon its fitness.

$$S_j^a(position) =$$

$$[\, S_j^{a-1}(position) + \ (mutation_{rate}) \times \ (|Fitness(S_j^{a-1}) - \ threshold\ |) \qquad (5.5)$$

In equation 4.5, $S_j^a(position)$ is the position of chromosome S_j in the iteration a, $mutation_{rate}$ and threshold are parameters of mutation process. And $Fitness(S_j^{a-1})$ is the fitness of chromosome S_j in the previous iteration (*a-1*).

The amount of mutation depends on the fitness of the parent chromosome. The fittest parent chromosome changes its position slightly, whereas the parent chromosome with the lower fitness changes its position drastically. In Figure 5-3, we demonstrate the pseudo code of GATSW. Figure 5-4 represents the process of GATSW.

The parameter threshold demonstrates the best similarity score that is equal to the fitness of perfect match. We use this parameter in mutation to control the rate of selection of children chromosomes for the next generation. The parameter $mutation_{rate}$ controls the influence of its fitness on the chromosome's position in the next generation. The more dissimilarity of the chromosome from the user defined subsequence results higher mutation. In this way, we simulate the sliding window process of the WTSW algorithm.

```
Step 1:Initialization
        I. Initialize the size of population and generation, and mutation rate;
        II. Generate initial population ;
Step 2: while(termnation condition){
          I. Evaluate the fitness of individuals using TSW;
          II. Select parent chormosomes;
          III. Mutate parent chromosome;
          VI. Select new population;
        }
Step 3: Pick the fittest chromosome as the final solution;
```

Figure 55-3. GATSW Pseudo code.

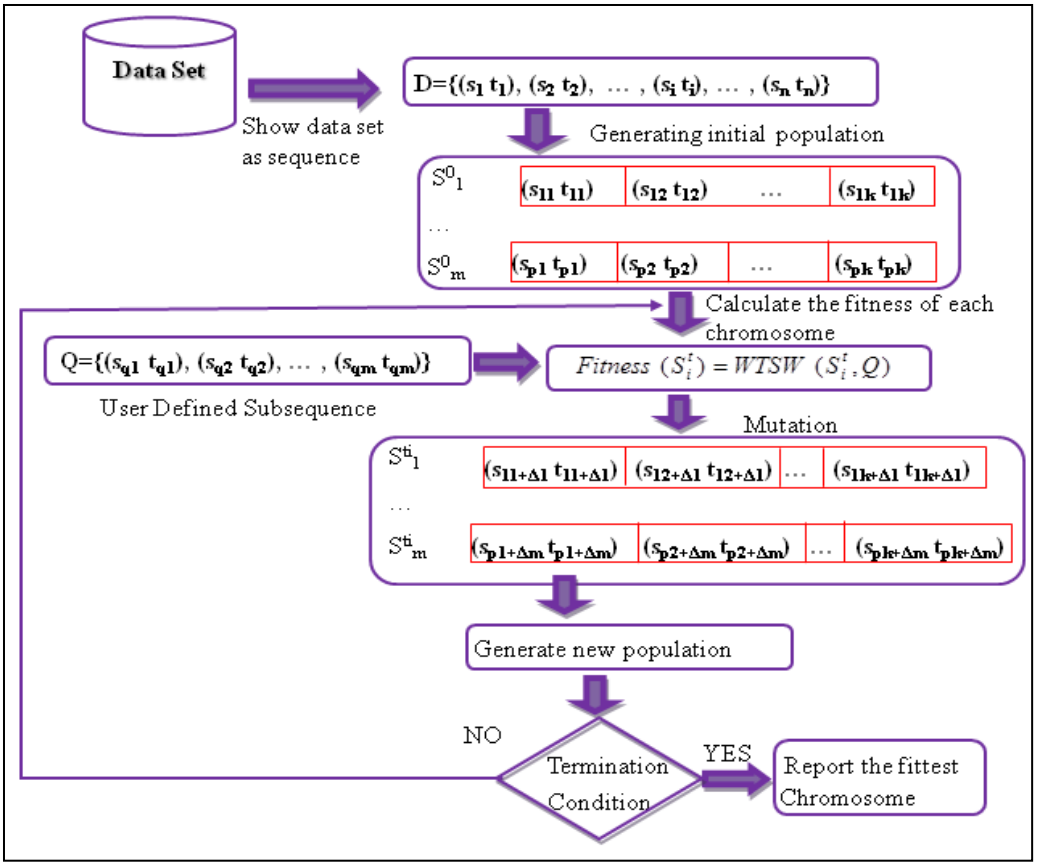


Figure 5-4. The flowchart of GATSW method.

Recall that in the WTSW, in each step we slide the window by W_Δ regardless of similarity of the current subsequence to the query subsequence. In the GATSW algorithm the amount of the sliding window is variable and depends on the similarity of the current chromosome to the query subsequence. For highly similar chromosomes the sliding window tends to 1 while for highly dissimilar ones it approaches *n* (the size of the query sequence). In this way, the GATSW algorithm guarantees that the genetic algorithm moves efficiently through the entire solution space.

5.3.4. Health Pattern Prediction

The health pattern prediction experiments are performed using a *k*-nearest neighbor and a leave-one-out cross-validation approach. For each unknown sensor sequence S_i, we compute the distances (using WTSW and GATSW for comparison) to all other sequences. Then we select *k* (*k*=9) most similar sequences together with their "normal" or "abnormal" labels. The sequences were labeled by clinicians by

retrospectively inspecting the nursing notes (see Table 5-1). Finally, the classifier predicts the label of S_i based on the label of its k nearest neighbors using the following heuristic: if any of the labels of the k retrieved sequences is "abnormal" then we label S_i as abnormal; if all k labels are normal, then S_i is labeled as normal. This heuristic was motivated by the fact that in medical applications the cost of a missed detection far outweighs the one of a false alarm. To evaluate our prediction algorithm, we used accuracy defined as:

$$accuracy = \frac{true\ positives + true\ negatives}{\#\ test\ data},$$

that is, the ratio of true positives and true negatives divided by the total number of data points in the test dataset.

5.4. Experimental Results

In this section, we first investigate TSW properties and compare it to LCSS on a synthetic data set. Then we demonstrate the performances of WTSW and GATSW methods on our TigerPlace dataset as well as a well-known benchmark dataset.

5.4.1. TSW Properties

To investigate TSW properties we conducted two experiments: one in which sequences have the same symbols but variable time stamps, and another in which the sequences are different but equidistant in time. We also compared the TSW with LCSS method.

5.4.1.1. Influence of Time between Symbols on TSW

To study the influence of the event time distribution, we used two different synthetic sequences of length 20, S_1 and S_2, to generate 10 patterns by changing the time stamp of the symbols (and keeping the symbols unchanged). Each pattern is 5 minutes long. The resulting 20 sequence set is denoted as $S_{\Delta t}$. The time stamp of each symbol was generated using a uniform distribution. Since we have 20 symbols distributed within 300 seconds, we expect an average time interval of about 15 seconds between events. We then

computed the pair-wise TSW similarity, M_{TSW}, between all $S_{\Delta t}[i,j]$ for all $i,j \in [1,20]$. As the first half of M_{TSW} was built based on S_1 and the second half based on S_2, we expect to see a clear separation between the two groups. To quantify the separation, we correlated M_{TSW} to the ideal similarity matrix IM_{TSW} that has two all 1 blocks of size 10 on the main diagonal and 0 in the rest. Figure 5-5.(a) shows the value of the Pearson correlation, $corr(M_{TSW}, IM_{TSW})$, for various values of parameter c in TSW. The best correlation (hence the best cluster separation) is achieved when c is close to the average time interval between the symbols, that is c=15s. Figure 5-5.(b) represents the M_{TSW} obtained for c=15s where we can observe a clear separation between the first 10 and last 10 sequences. This experiment provides a method for computing the constant c in equation 3 for other sequence types.

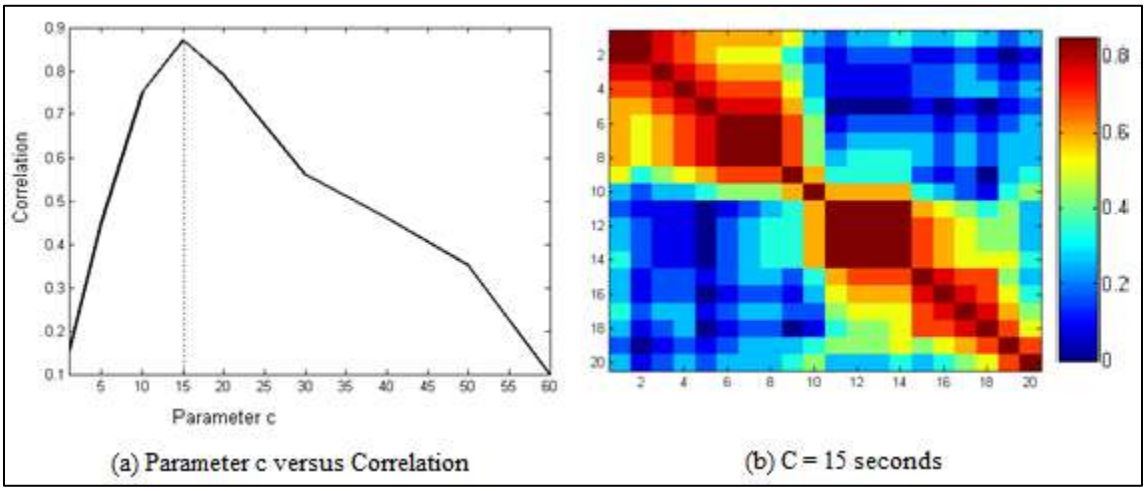


Figure 5-5. The effect of the parameter c on Synthetic dataset.

5.4.1.2. Influence of the Rate of Symbol Change on TSW

In this experiment, we used a synthetic dataset of 20 different sequences with equidistant time but different symbols, to get a sense of the TSW sensitivity to symbol change (mutation). In a similar fashion as above, we generated 10 sets of 20 sequences (length 20) starting from two different sequences, S_1 and S_2, by changing 1 to 10 symbols at a time (mutation rates between 1/20=5% to 10/20=50%). We also used the Pearson correlation between the pair-wise similarity matrix, M_{TSW}, and the ideal one IM_{TSW} to assess the cluster separation. Figure 5-6.(a) shows the value of the correlation for various mutation values. Figure 5-6.(b) depicts the similarity matrices *MTSW* for mutation values

2 (2/20=10% mutation rate) and 5 (5/20=25% mutation rate). We see that even at 25% symbol difference (Figure 5-6.(b) right side plot), the similarity between two sequences is significant (about 0.6). The figure shows the power of the TSW as similarity measure for multi-dimensional time series. Next, we compare the TSW to a well-known method, LCSS [27].

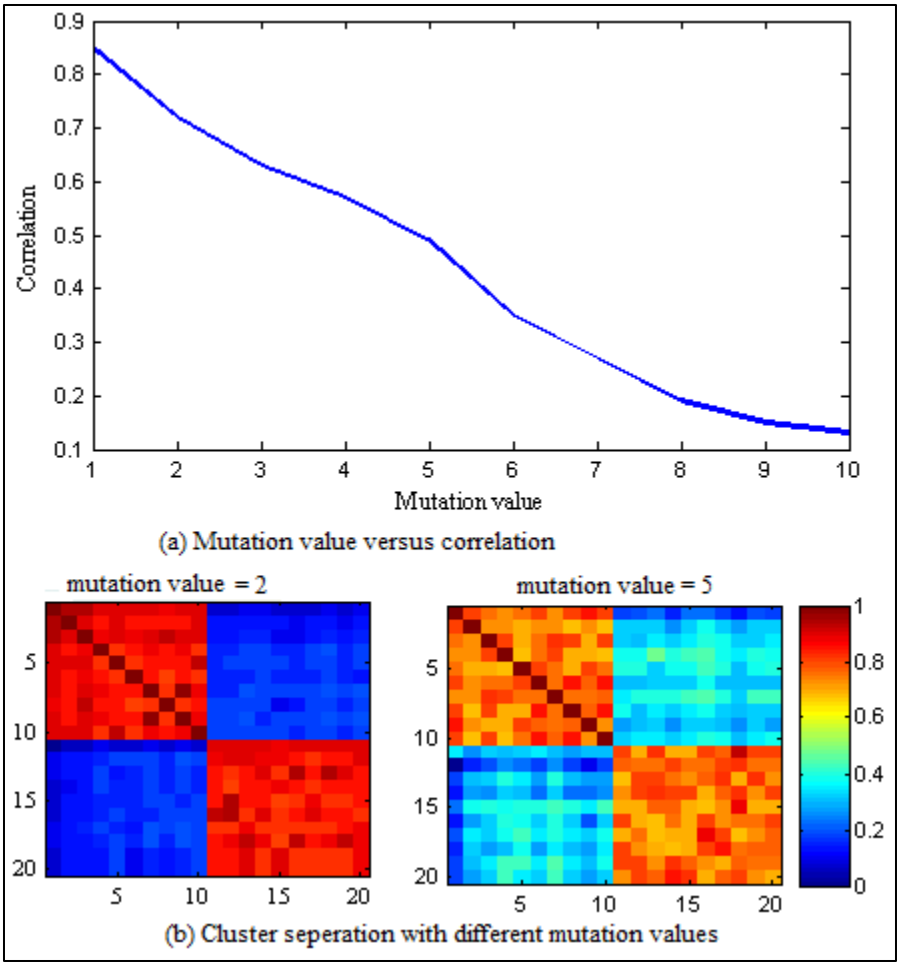


Figure 5-6. The effect of mutation value on similarities in synthetic dataset.

5.4.1.3. Comparing TSW and LCSS methods

To evaluate the performance of TSW we compared it to LCSS on the synthetic dataset $S_{\Delta t}$ described before. LCSS is a similarity measure between two sensor sequences in the multidimensional time series context [27]. Let $S_1 = \{(s_1\ t_1), (s_2\ t_2), \dots, (s_n\ t_n)$and $S_2 = \{(s'_1\ t'_1), (s'_2\ t'_2), \dots, (s'_m\ t'_m)\}$ be two sequences of length n and m respectively.

We define $Head(S_1) = \{(s_1\ t_1), (s_2\ t_2), \ldots, (s_{n-1}\ t_{n-1})\}$. The following formula describes LCSS:

$$LCSS(S_1, S_2) = \begin{cases} 0 & if\ S_1 or\ S_2\ is\ empty; \\ 1 + LCSS_{\delta,\varepsilon}\big(Head(S_1), Head(S_2)\big) & \\ \qquad if\ |s_1 - s_n| < \varepsilon\ and\ |t_n - t'_m| < \delta; & \\ MAX\left(LCSS_{\delta,\varepsilon}(S_1, Head(S_2)), LCSS_{\delta,\varepsilon}(Head(S_1), S_2)\right), & otherwise. \end{cases}$$

Having two sequences S_1 and S_2, the idea of LCSS is to count the number of pairwise matches in sequence T_1 and T_2 with respect to the user-defined thresholds δ and ε. The parameter ε is a matching threshold whereas δ controls an acceptable time difference for matching a pair of points in two sequences T_1 and T_2. The effect of threshold δ in LCSS is somewhat similar to the effect of c in our similarity. We compared TSW and LCSS on $S_{\Delta t}$ for various values of c and δ by calculating the Pearson correlation between the pair-wise similarity matrix, M_{TSW} for TSW and M_{LCSS} for LCSS, and the related ideal ones, IM_{TSW} and IM_{LCSS}. Figure 5-7.(a), shows the correlation $corr(M_{TSW}, IM_{TSW})$ for different values of parameter c (solid line) and the correlation $corr(M_{LCSS}, IM_{LCSS})$ for various values of parameter δ (dashed line).

This plot shows that TSW outperforms LCSS for every value of c and δ, respectively. In Figure 5-7.(b) we show the cluster separation for similarities TSW and LCSS and c, δ=15. Arguably, the TSW similarity matrix (right) shows stronger cluster separation than the LCSS one (left).

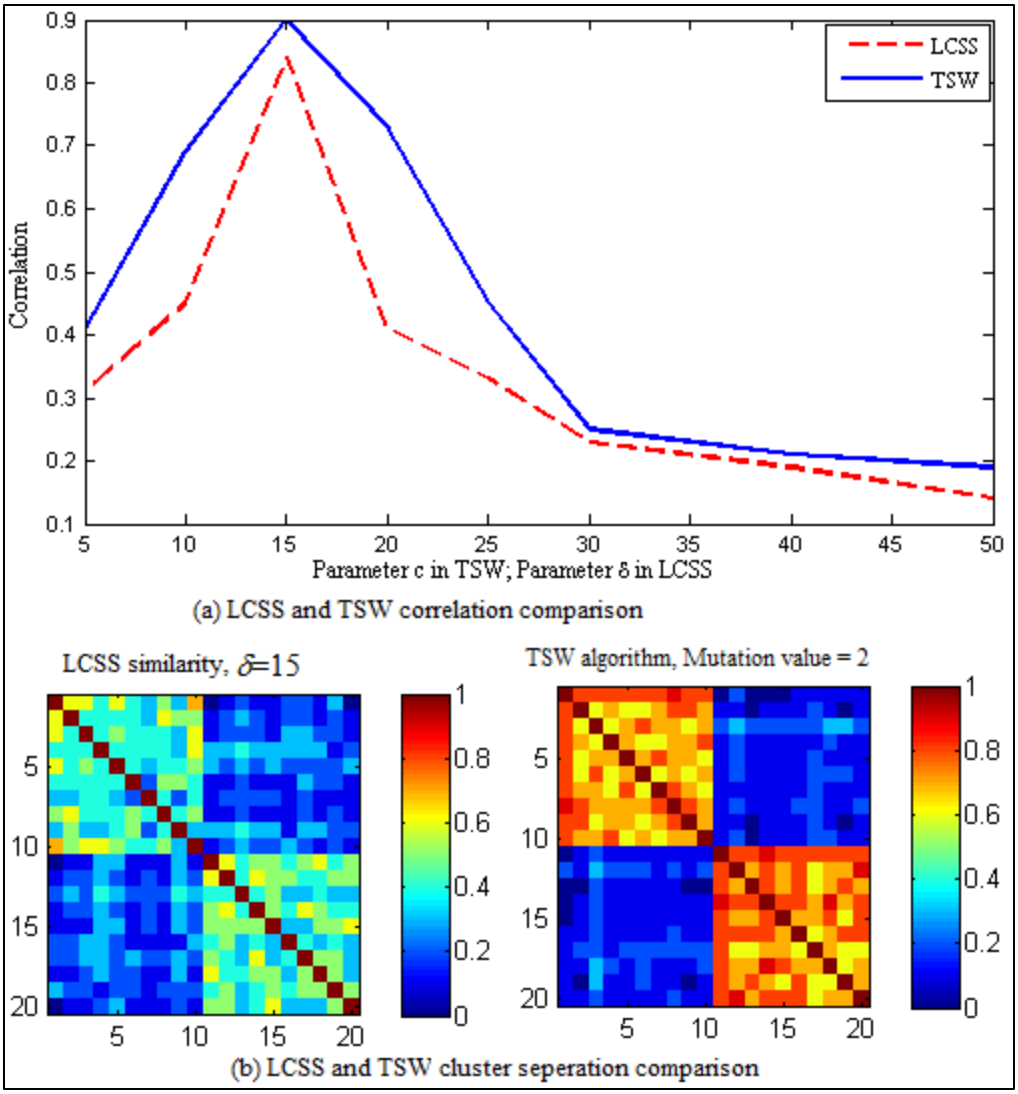


Figure 5-7. Comparing the performance of TSW and LCSS methods.

5.4.2. TSW, WTSW, and GATSW analysis

We consider the problem of finding the most similar multi-dimensional sensor time series to a user defined sensor sequence. In Figure 5-8, we show 3 visualization modalities for a sensor sequence. In Figure 5-8.(a) the sequence is represented by the name of each sensor and its firing time. To reduce the required space, in 5-8.(b) the sensor name is abstracted to a letter of the alphabet. To visualize long sequences (Figure 5-8.(c)) we use an indexing of the sensor names to a color map where each color level represents one sensor, and a white means no sensor firing. We use this data visualization to better understanding our data and simplify the validation of experimental results.

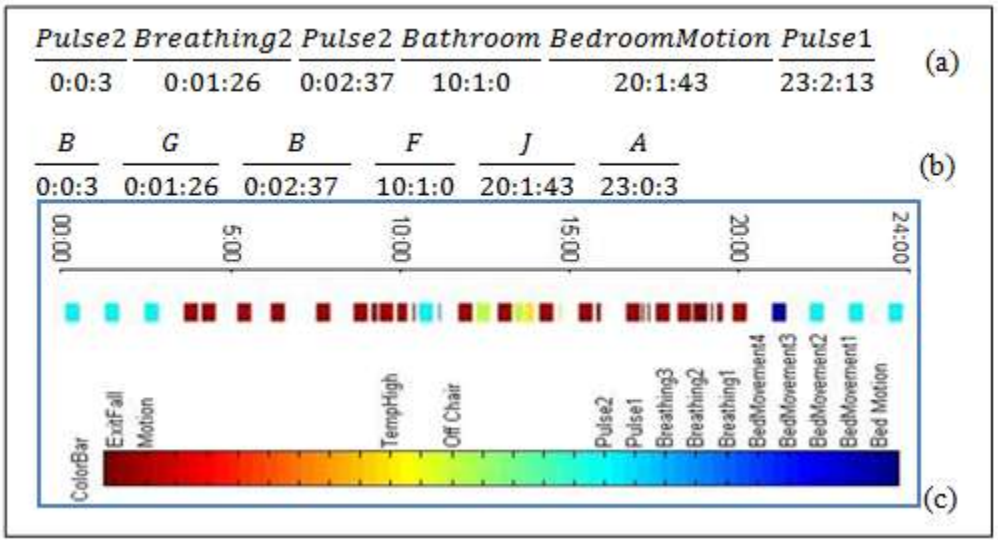


Figure 5-8. Possible visualization modalities for a sensor sequence.

5.4.2.1. Finding the value of parameter *c* in TSW

An important constant that we need to determine is the temporal gap parameter *c* in TSW. For this purpose, we compute the sensitivity of classification of each subsequence as normal/abnormal day using our dataset shown in Table 2-1. The variation of sensitivity with parameter *c* is shown in Figure 5-9.

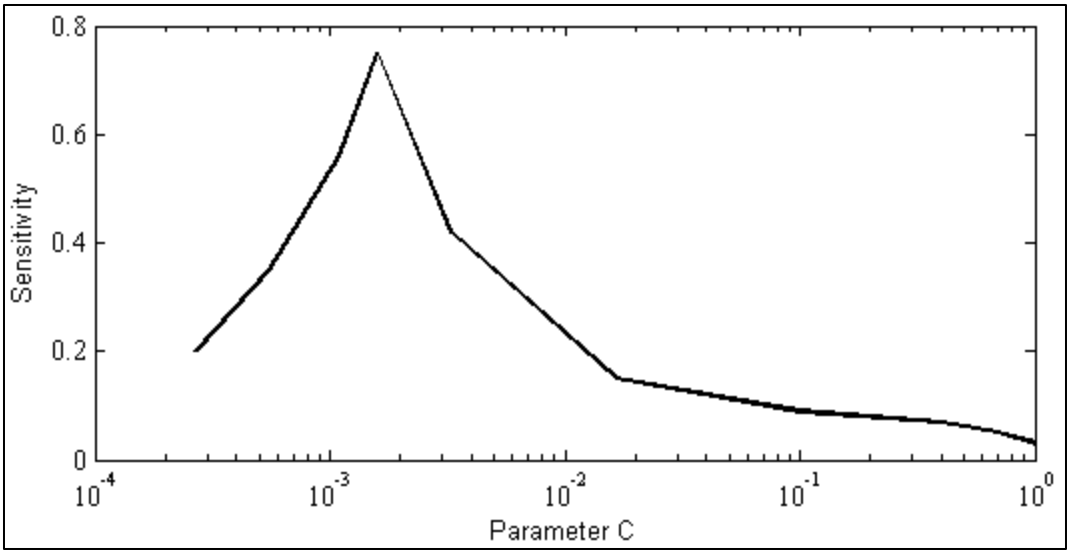


Figure 5-9. The effect of parameter c on the sensitivity of classification.

We observe that the sensitivity has a sharp maximum around c=0.002 which means that the relevant time interval is 500s~8 minutes. We averaged the results for all of our residents. This is not unexpected since for elderly population, in normal days, important daily events such as going to bed, waking up, bathroom visits, meals, etc. tend to happen roughly at same time of the day(+/- one hour). For this reason, we will perform the rest of our experiments with c=0.002 and g=0.

5.4.2.2. Finding best value of W_Δ in WTSW

In the WTSW algorithm, we need to choose the sliding window size, W_Δ. This parameter has an important effect on the quality of the results: a small W_Δ value increases the time complexity whereas a large W_Δ value doesn't guarantee reaching to the best possible solution.

In this section, we present a set of experiments to evaluate the effect of parameter W_Δ on execution time and accuracy of the WTSW similarity measure. Figure 5-10 shows the execution time in milliseconds and the average of similarity of top 10 subsequences based on different initial values of sliding window (W_Δ). The experiments were run independently for each resident (see Table 2-1) and then the results were averaged across residents.

By increasing W_Δ the execution time decreases significantly. This is because larger W_Δ results in fewer window candidates and requires less calculation time by the WTSW

method. In this set of experiments W_{Δ} is expressed in minutes. By increasing W_{Δ} from 5 to 50, the execution time declines significantly from 2066 ms to 581 ms. Moreover, as we increase W_{Δ} the similarity score decreases. This is because the algorithm discards some possible subsequences which may contain better solutions. Considering the results of this experiment, we decided to initialize W_{Δ}=15 minutes where we lose about 3% in similarity but increase the speed by about 300%.

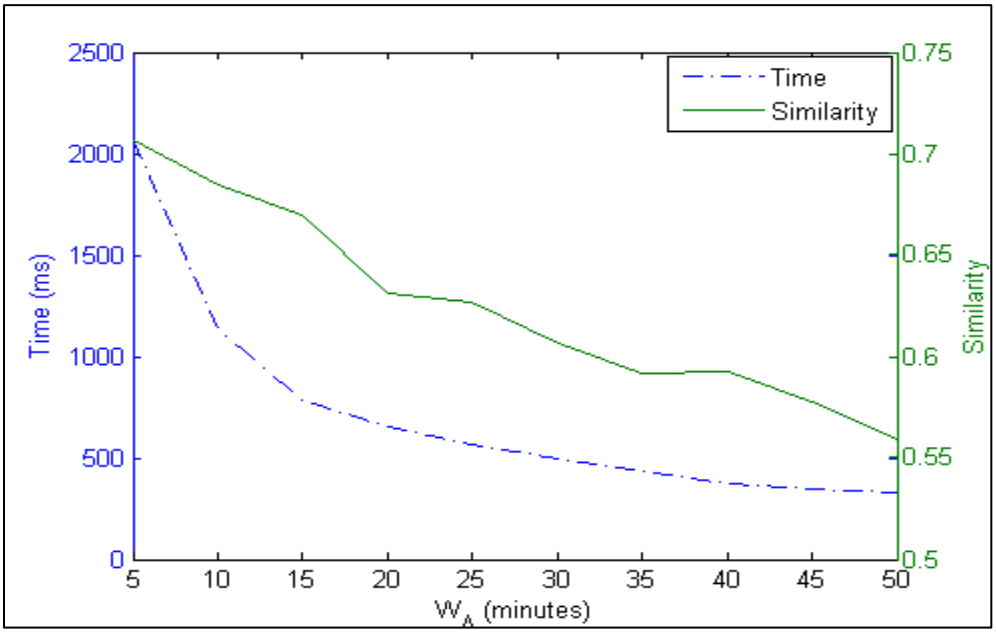


Figure 5-10. The effect of parameter W_{Δ} on time and similarity in WTSW.

5.4.2.3. Finding best value of size of population in GATSW

The size of population controls the number of chromosomes (candidates) in the GATSW algorithm, relation depicted in Figure 5-11.

This experiment demonstrates that by increasing the size of population the average fitness of the top 10 candidates improves from 0.74 to 0.87 due to the increased chance of finding the global maximum solution. However, increasing the population size over 60 chromosomes decreases GATSW performance by 2% and increases the execution time by 50%. To balance performance and execution time, we set the size of population to 60 for the rest of our experiments.

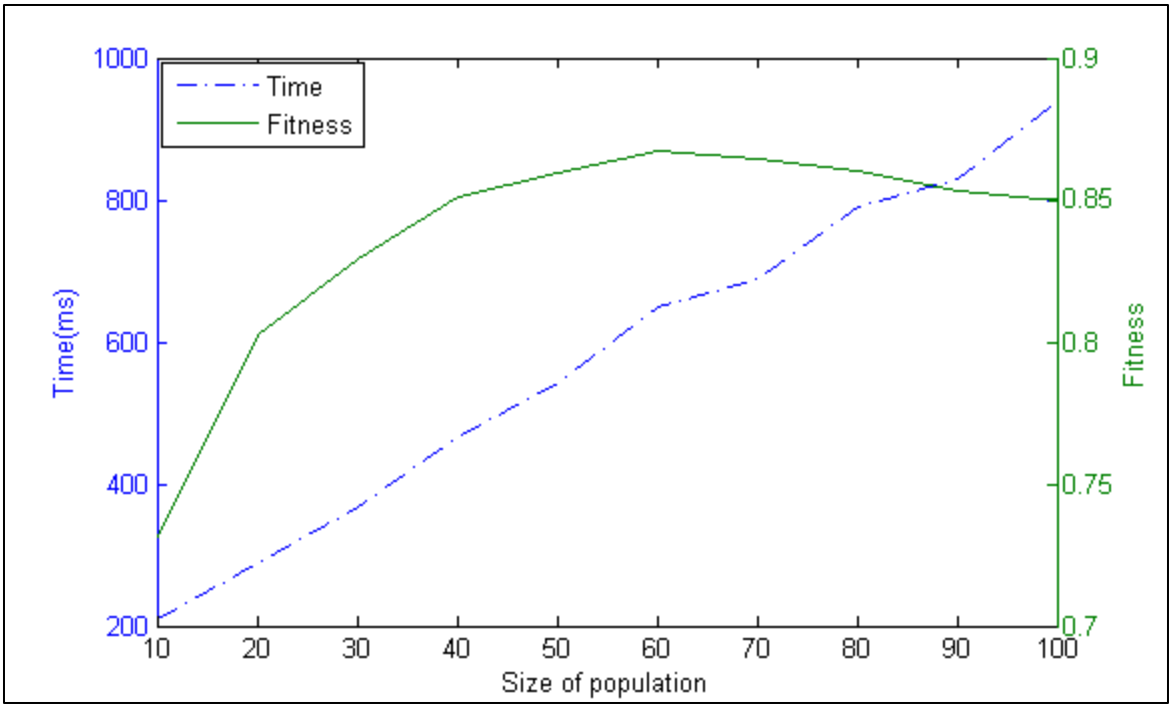


Figure 5-11. The size of population versus fitness and execution time in GATSW.

5.4.3. Health Pattern Detection Using WTSW and GATSW

Our goal is to build an early illness recognition system for the TigerPlace residents using sensor pattern similarity. For a given resident with available historical sensor data, the EIR system searches for similar patterns to the current recorded activity. The high similarity of the current sensor pattern to one in the past that was related to an abnormal health event indicates the possibility that the previous illness reoccurred. In the next two sections, we analyze the performance of WTSW and GATSW in health pattern recognition in terms of retrieving "normal" and "Abnormal" days.

5.4.3.1. Health pattern detection using WTSW

The length of motion activity in time (t_{Δ}) is an important factor in the performance of the monitoring system using WTSW. We show this paradigm in Figure 5-12. In this experiment, the scale of t_{Δ} is hours. A t_{Δ} of size 2, for example, shows an activity of a resident within a window of two hours. In Figure 5-12, we represent the effect of t_{Δ} on the accuracy of health pattern detection and the execution time using WTSW method.

Generally, by increasing t_{Δ} the accuracy decreases around 12%. The reason is in a larger time interval it is less likely that residents do the exact same thing in the exact same time of every day. Since the similarity measure is impacted by time stamp, the time of a day when an action is taken affects the similarity values and so the accuracy of prediction. An exemption can be a night time interval when residents are more likely asleep and the activity patterns should be almost the same in the most of days. This is an important feature of the WTSW method, since in clinical perspective same activities in different time of the day should be considered differently. For example mid night bathroom visits are studied differently than day time bathroom visits.

In terms of speed, by raising t_{Δ} the execution time increases almost linearly as expected since TSW complexity is $O(mn)$. The increase in execution time due to an increase in t_{Δ} differs for each resident depending on the sensor firing density. In average, increasing t_{Δ} from 1 to 10 hours, increases the execution time about 6 times (from 2 to 12 s for our dataset) which might become a problem when the algorithm is run hourly for thousands of patients. To speed up the WTSW we employ GATSW, which we present next.

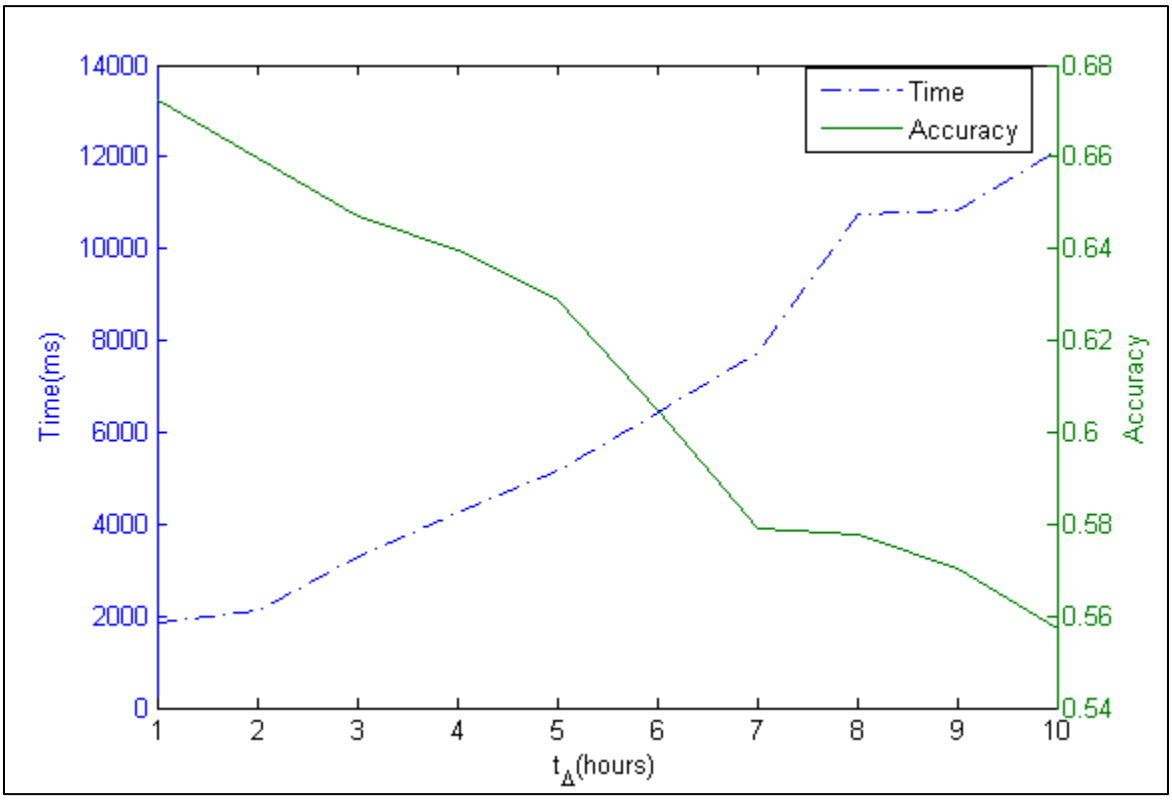


Figure 5-12. Health pattern detection based on t_{Δ} using WTSW.

5.4.3.2. Health pattern detection using GATSW

In the genetic algorithm, initializing the number of generations highly affects its performance. In this experiment (Figure 5-13) we investigated how different values of the number of generation influences the performance of GATSW for pattern prediction. Figure 4-13 also represents the execution time based on the number of generation. Clearly, increasing the number of generations requires more execution time. Increasing the number of generations from 10 to 100, results in a 6 times rise (from 1 to 6 s for our dataset) in the execution time which is two times faster than WTSW method. This also affects the accuracy of health pattern prediction. As we increase the number of generations the prediction accuracy raises, but it has a decline after 60 generations. This is probably because the number of generations is too large that the algorithm falls in a local optimum. The best accuracy, 0.72 was obtained for a population size of 60.

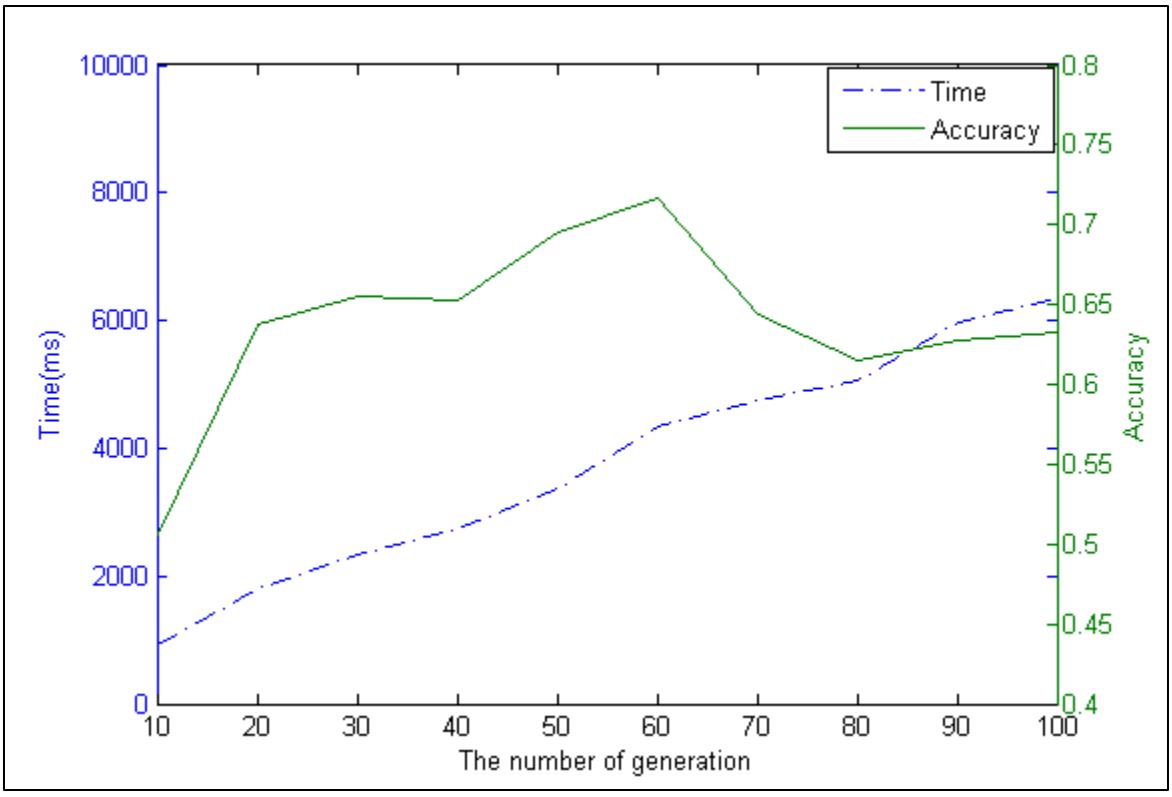


Figure 5-13. Health pattern detection based on the number of generation using GATSW.

5.4.4. GATSW Evaluation on a Benchmark Dataset

For further evaluation, we applied our proposed methodology to the localization data for posture recognition dataset [15]. The dataset contains the motion sensor hits of five people performing 11 activities such as walking, sitting, falling, lying down, standing up from sitting on the ground, lying, sitting down, standing up from sitting, standing up from lying, on all fours, and sitting on the ground. Each person performs these activities in five different episodes while wearing four motion sensors on the left ankle, right ankle, chest, and belt. In average, each episode lasts 3 minutes, and between two consecutive episodes there is a roughly 3 minute pause (no activities). The dataset contains a total of 164860 sensor hits, and the stamp time is given in milliseconds. The average time difference between two consecutive sensor hits is 32 milliseconds. We run a set of experiments to find the best value for the parameter c in Eq.4.3. We observe that the sensitivity of the posture recognition has a sharp maximum around c=0.0003 which means that the relevant time is 2750 milliseconds~2.7 seconds. We evaluate our experiments using, *precision*, *recall* and *F-measure*.

For posture recognition, we use a k-nearest neighbor classifier (k=7) using GATSW as similarity function with a leave-one-out cross-validation approach. Figure 5-14 shows the *F-measure* of the posture recognition and execution time using GATSW method for various t_{Δ}. The best *F-measure* that GATSW achieve is 0.82 using a window of t_{Δ} =3 seconds. In [15], authors used a fusion of SVM and C4.5 algorithms and reported an overall *F-measure* of 72%. Our proposed method improves the posture recognition on this dataset by 10%.

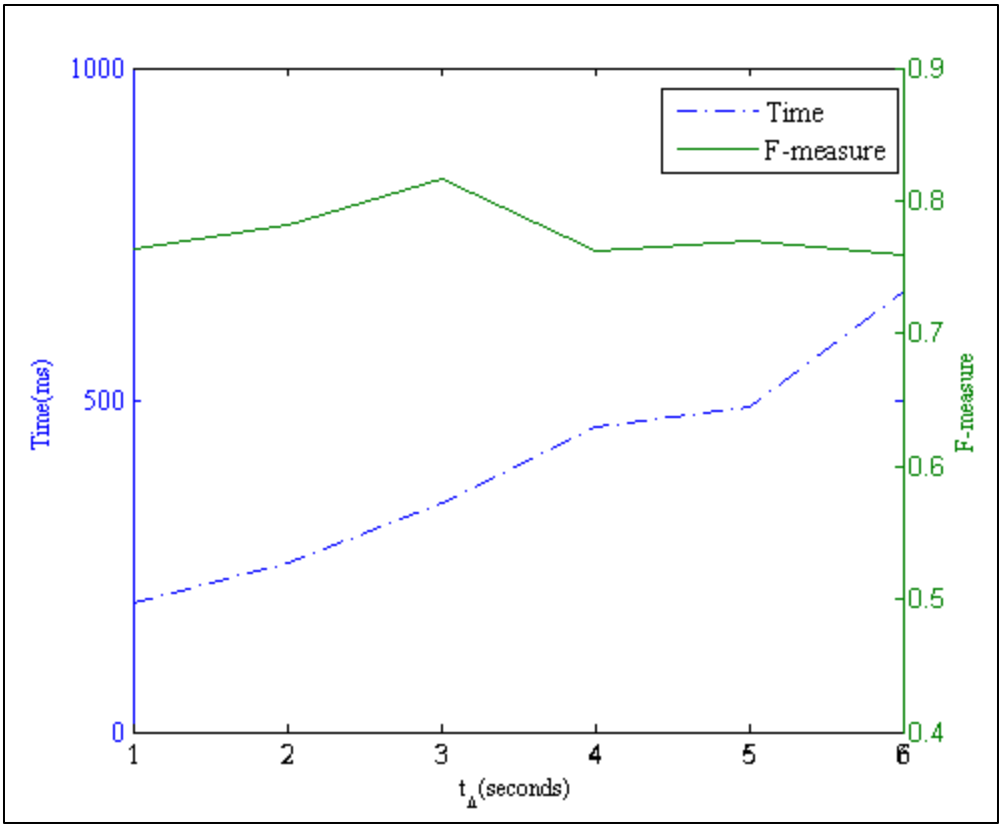


Figure 5-14. Effect of window size on GATSW for benchmark dataset.

5.5. Conclusion

In this chapter, we proposed a similarity measure, TSW, for MATS. Then, we introduced a window based algorithm, WTSW, which uses TSW to search for the best match in long MATS. Since WTSW could be potentially slow for eldercare applications, we proposed a genetic version of it, GATSW. Finally, we demonstrated how TSW can be used in a framework for detecting health patterns.

We tested our algorithms on multiple datasets: two synthetic ones, one obtained in TigerPlace and another one obtained from [15]. On the TigerPlace dataset, we obtained an abnormal day prediction accuracy of about than 70%. On the sensor dataset obtained from [15] we obtained about 82% F-measure on posture recognition, which is 10% better than the results published in [15].

In the next chapter, we present a new early illness recognition approach based on TSW pattern similarity. We use TSW and introduce a new confidence level for the purpose of classification.

Chapter 6 Detection of Abnormal Sensor Patterns using TSW

The aim of this study is to predict the change in resident's health status based on sensor data produced by in-home monitoring system. The power of sensor networks for predicting health patterns using logistic regressions, one class classifiers, multiple instance learning and temporal clustering has been previously investigated [127]. In chapter 2, we proposed a frame work that uses Euclidian based methods (such as RMS) to find similar sensor sequences and detect abnormal events. Euclidian based dissimilarity measures are sensitive to noise and outliers. To overcome this challenge, we tested non-Euclidian based measure such as DTW for health patterns detection. The performance of this framework was not desirable because of the data conversion mechanism. Aggregating sensor data time-wise and sensor type-wise results information loss and low precision.

In chapter 4, inspiring from bioinformatics techniques we proposed a new similarity function (TSW) that avoids data aggregation. TSW considers sensor sequences as a time series of pairs of sensors and their time stamps. It uses a dynamic programming approach to locally align two sensor time series and calculates their similarities. In chapter 5, we analyzed the performance of TSW as similarity function and proposed two modified version of it as WTSW and GATSW. The WTSW relaxes the condition of TSW on the length of sensor sequence, whereas the GATSW enhances the speed of WTSW using a genetic algorithm approach. We evaluated the performance of TSW, WTSW, and GATSW in a *k*-NN classifier to predict abnormal sensor events.

In this chapter, we propose a new approach that uses TSW as similarity function, and introduce a new confidence level for the purpose of classification. In section 6.1, we provide an introduction and present a review on related literatures. In section 6.2, we describe our method. Section 6.3 shows the experimental results and discussion.

6.1. Introduction

In TigerPlace, our goal is to help the residents not only manage their illnesses but also stay as healthy and independent as possible. We used sensor technology and implemented a framework to provide early identification of problems in mobility and

cognition. Sensors provide valuable data for the assessment of volumes, intensity and patterns of activity (PA) across the daily and weekly patterns [142].

The aim of this research is to find the connection between the change in health status and motion sensor data produced by the in-home monitoring system. The primary goal is to use data mining and machine learning techniques to find a link between abnormal levels of daily activities and vital signs of residents. An abnormal bathroom visits pattern and/or low level of motion may be related to urine infection or an enlarged prostate gland. Our sensor data capture motion activities about the resident. By continuously computing the motion activities and comparing it with the measured trend we may alert the nursing staff when some predefined variability limits are exceeded. This approach may provide additional activity monitoring for the elderly such as variations in the length of bathroom visits during the time between two nursing visits. In addition, the comparison of the computed and measured bathroom visits trends over longer periods of time may provide additional warnings of abnormal unreported clinical events. The algorithms will be validated using the clinical data of the participants connected to the in-home monitoring system.

To detect changes in motion activities and recognize an abnormal event the primary task is to consider the history of elderly resident's motion activities. Intelligent sensor data analyses techniques group sensors based on domain knowledge and specific activities of interest. Each ADL is associated with a group of sensors which fuses data from all of sensors in that group related to that activity. In [143], authors suggest an efficient approach that rather than gathering data from all sensors and carrying out an overall data-mining algorithm focuses on each activity according to the available knowledge and the data collected by the corresponding sensor group. They considered three types of activities in the analyses phase: long-term trends, significant patterns, and associations among patterns.

The analysis of well-being is a challenging task because of difficulties to identify abnormal behavioral patterns. In fact, a behavior that is considered normal for one person might be identified an abnormal event for another person. Additionally, people change their manners and ways to do their things without necessarily being affected by deterioration in their physical or mental abilities. For example, weather conditions highly

affect the way people performs their daily tasks. Therefore, interactive and adaptive algorithms are necessary to handle such analysis with the particularities of each individual in mind.

From the computational perspective, various data mining techniques have been published in literature. Decision trees are used as classification algorithm in [144], [145]. Authors applied pruned C4.5 decision tree algorithm because of accuracy and simplicity of the generated rules. Moreover, other computational algorithms has been studied such as SVM [145] [146], logitboost that uses logistic regression as the cost function in AdaBoost [145], rule based approach [147], mixture models as Gaussian mixture [148], pattern recognition algorithms [149], sophisticated data mining techniques and machine learning algorithms like Markov chain [148], artificial neural networks [150], and Bayesian models [151].

In [152], authors proposed the application of frequency and rank order statistics for monitoring mobility changes of the elderly in their residence. They discriminated patterns generated from healthy and pathological states since complex physiological signals may carry unique dynamic signatures related to their underlying mechanisms. They proposed a data fusion framework based on evidence or probabilistic theory [153] that combines health status or autonomy, heterogeneous data coming from multi-sensor acquisitions to produce a better estimation of the activity.

A common limitation to all of these systems is the small number of dataset in term of the number of subjects participating in implementation or validation phases which complicates extrapolating results. Another common problem is the very short series of testing [142] [152]. Using simulated data of different behaviors on long or short periods is also another limitation of reported studies [153].

To address these limitations and increase the accuracy of the models used in processing sensor data, some of the authors proposed several solutions of improvement, such as using more powerful techniques like SVM [148], a finer granularity of data [149], fusion of data coming from several sensors [153] [154] or identifying and using an optimal size of data sets [144].

6.2. Activity detection using distribution of normal events

In chapter 4, we introduced a new multi-dimensional sensor sequence similarity, TSW that considered sensor patterns as 1-dimensional sequences of characters together with their associated time stamps. In TSW, the time stamp shows the time when a character (firing) is emitted by the related sensor network.

Essentially, TSW considers the time of the day between the sensor firings as a gap and computes the gap penalty $W_{\Delta t}$ by using time stamps (see equation 4.3). We use the "time of the day" metric, since we would like to find similar behaviors across different days that happen at roughly the same time. Note that the type of time used in (see equation 4.7) is critically important and depends on the application. In our case, the time in (see equation 4.7) is input in seconds. In this paper, we use TSW to classify sensor sequences as "normal" or "abnormal" days. "Abnormal" days are defined by unusual sensor activity patterns that require a nurse's assessment of the resident and were documented in EHR. The system automatically sends an email alert to the nurse in case that some unusual pattern is detected. After assessing the resident, the nurse records pertinent comments in the EHR. However, on a "normal" day the sensor activity pattern is not flagged to prompt a nurse's visit to the resident. We note that a simplified alert system based on individual sensor values is already in place at TigerPlace. The disadvantage of this system is that a given medical condition generates multiple alerts (one for each sensor) that may confuse the clinical staff. Instead, our system generates only one alert per incident.

6.2.1. Abnormal pattern detection algorithm

Given n (training) normal sensor sequences $\{S_i\}_{i=1,n}$ we compute the pair-wise similarities between them, $\{s_{ij}\}_{i,j=1,n}$ using TSW. We, then, calculate the distribution of the $\{s_{ij}\}$ similarities of these "normal" days assuming they follow a Gamma distribution. Assume that we found out that $\{s_{ij}\}$ follows a distribution Gamma with parameters a and b, i.e. $\Gamma(a,b)$. To classify an unknown sequence S_x, we start by computing its similarities with all normal sequences $S_1,\ldots S_n$, obtaining similarities $\{s_{ix}\}_{i=1,n}$. Then, we find the maximum of $\{s_{ix}\}$, $s_{x,max}$. The confidence that S_x is abnormal, $C(S_x)$ is 0 if

$S_{x,max}$>*mean(Γ(a,b))* and equal to 1-*P* where *P* is the likelihood that $s_{x,max}$ comes from *Γ(a,b)* and is calculated using:

$$P(s_{x,max}, a, b) = \int_0^{S_{x,max}} \left(\frac{1}{\Gamma(a)b}\left(\frac{t}{b}\right)^{a-1} e^{-\frac{t}{b}}\right) dt \quad (6.1)$$

While in our experiments we use all normal days available, in a real system implementation we would use the data from the last two weeks.

6.2.2. Classification experiments

The classification experiments are performed using a leave-one-out approach. We compare our results to the ones obtained using a *k*-nearest neighbor (*k-NN*) approach with *k=1* and using the same TSW distance as above. In the next section, we present our experimental results.

6.3. Experimental Results

6.3.1. Data Set

To investigate the performance of the proposed illness prediction methodology based on TSW, we run a set of experiments to train a classifier for each resident separately. We used the same sensors presented in Table 2-1. We processed a log file of the sensor events (displayed in the Table 2-2) to extract the sensor sequences of "normal" and "abnormal" days in a predefined time interval. For this study we used the pilot sensor data from the apartments of three TigerPlace residents (see Table 2-4). For each resident, we also retrieved visit notes about physical, emotional and other health complaints, recorded by the nurses in the TigerPlace nursing EHR.

6.3.2. Classification Results

We initialized the parameters of TSW as *g*=0 and *c*=0.002 which means that the relevant time interval is about 500s~8 minutes (see section 3.3.1). Figure 6-1 shows similarity distribution in the morning time interval (8am to 10am). "Normal" day's similarity has a Gamma distribution. We used this distribution to set the threshold in the training classifier phase. From this point of view our method represents a one class

classifier: if the similarity between a new day and the previous normal ones has a low likelihood, then the day is either abnormal or represents a new pattern.

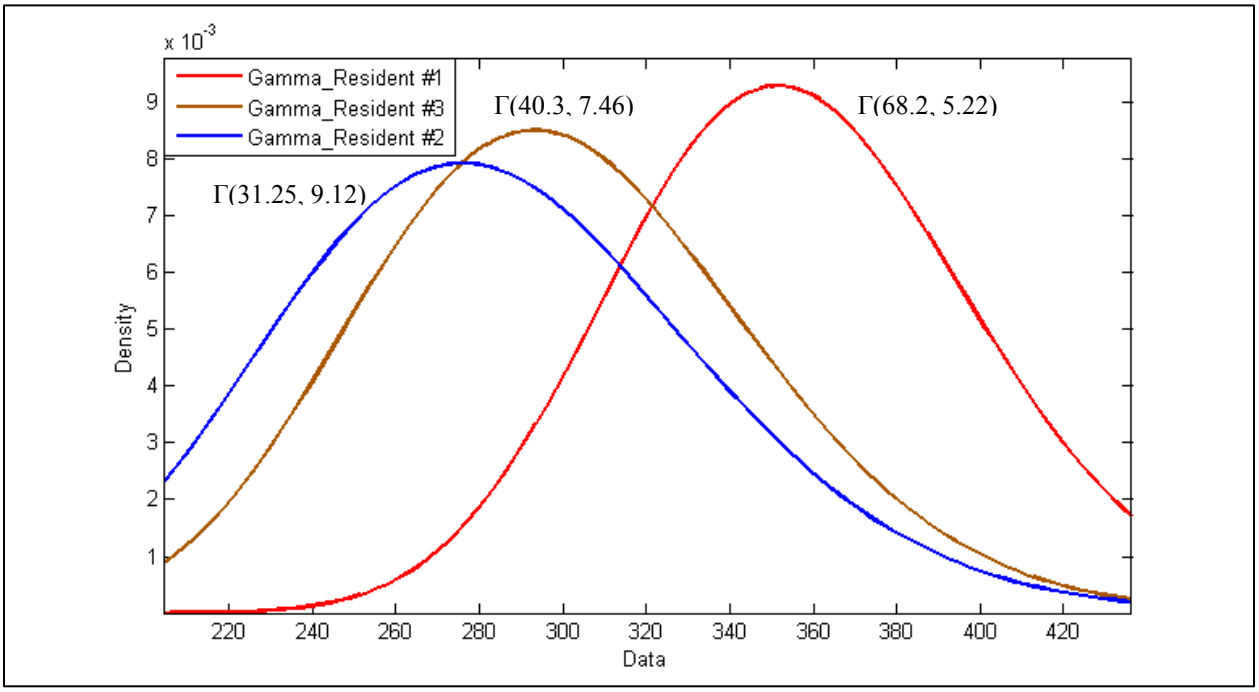


Figure 6-1. Distribution of normal day pattern similarity.

For similarity scores less than the threshold, we used equation (6.1) to calculate the confidence level of the similarity between two sensor sequences. This formula inspires the Gamma probability distribution to consider not only similarity scores but also the density of the scores of similar days. We used this similarity score and confidence level to train a classifier. For comparison purposes, we run a set of experiments with one class *k*-nearest neighbor (*k*=1) classifier where we use the similarity score without confidence level. We show the results of this comparison in Table 6-1 as area under ROC curve. Our proposed method outperforms *k*-nearest neighbor (*k*=1) in terms of accuracy. The main reason for this outcome is in *k*-nearest neighbor (*k*=1) method we do not consider the distribution of the similarity scores. In elderly health monitoring systems, usually emergency situations are so different from the elderly health history. Therefore, classifying based on the similar abnormal patterns does not produce satisfactory results. However, TSW algorithm with distribution similarities considers the history of normal patterns and using a confidence level classifies all abnormal patterns in a more accurate

way. Consequently, this method is more accurate compare to *k*-nearest neighbor (*k*=1). We present the area under the curve (AUC) curves comparisons in Figure 6-2.

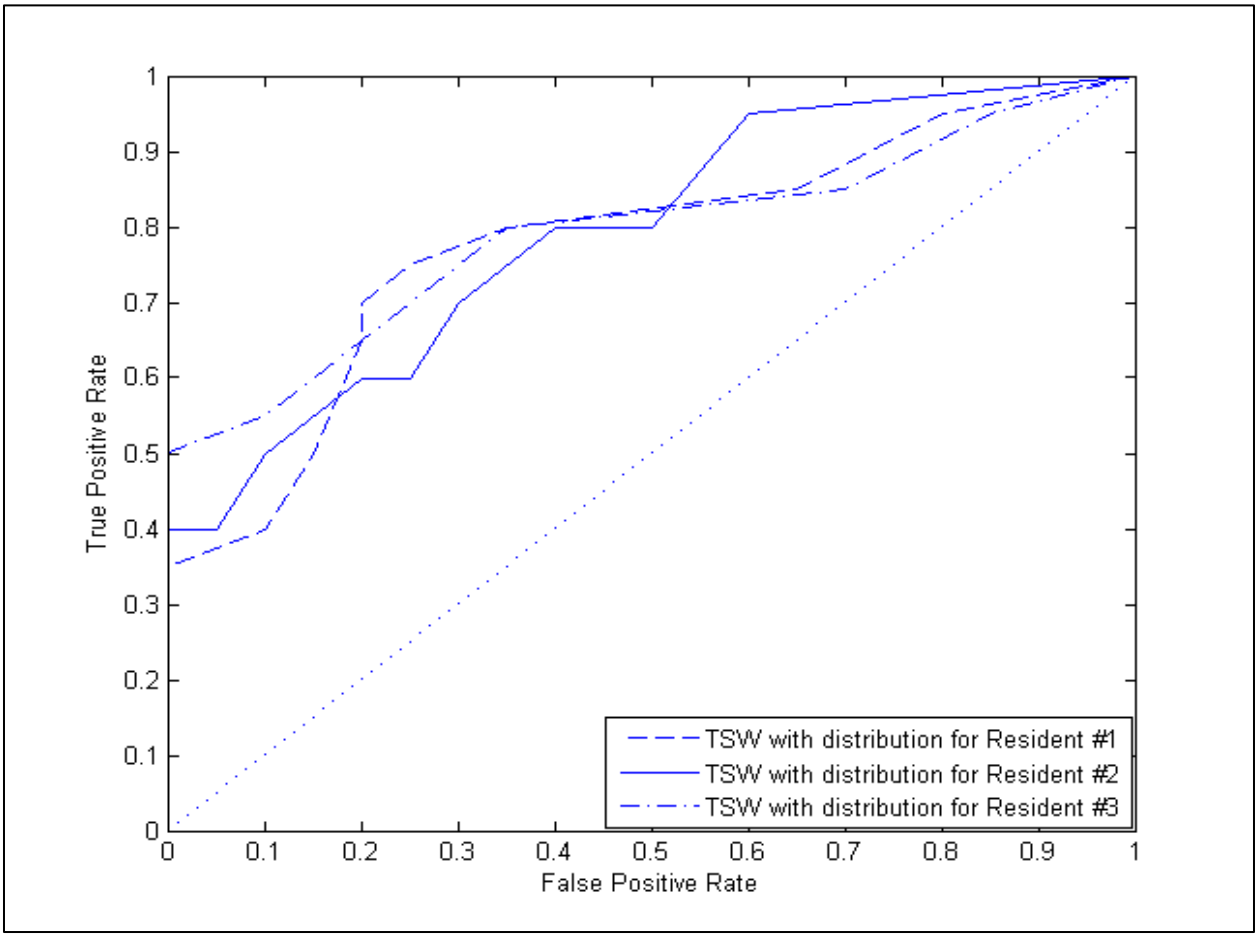


Figure 6-2. Comparison between the results obtained using the similarity distribution and *k-NN*.

Table 6-1. Comparison on the performance of methods based on AUC.

Similarity Method	**AUC #1**	**AUC #2**	**AUC #3**
TSW with Distribution	0.79	0.78	0.79
TSW with *k-NN*	0.54	0.56	0.60

6.4. Conclusion

In this chapter, we enhanced our framework for illness prediction using sensor networks. We used TSW as similarity measure to train a *k*-nearest neighbor classifier. We compared our results with binary *k*-nearest neighbor. In this comparison, the proposed TSW framework outperformed binary *k*-nearest neighbor. In the next chapter, we describe a novel approach that predicts the possible health issues based on sensor data and nursing notes produced in TigerPlace. We explain the proposed semi-supervised clustering method that automatically uses user provided information to effectively cluster nursing notes.

Chapter 7 Sensor Sequence Annotation Using TSW and NLP

Prevention saves billions of dollars each year for the healthcare industry. Early recognition of illness signs allows healthcare providers to promptly intervene and anticipate future health changes. To improve efficiency of an automated monitoring system, it is crucial to not only detect but also annotate clinically significant events with the possible causes of health issues. However, this process is costly in terms of time and manpower because it requires manual labeling by a medical expert. We propose a new semi-supervised learning framework for an automated early illness recognition system.

We design a novel framework that detects change in an older adult's activity from sensor data and predicts the possible health issues based on the existing nursing notes within our integrated in-home monitoring system. We propose a semi-supervised clustering method to effectively group nursing notes based on user-provided information. This framework represents each cluster by a language model and uses a bi-gram model of terms to improve the efficiency of the proposed clustering method. We estimate the cluster model by a maximum a posterior probability to address the challenge of imbalanced dataset clustering.

Our system achieves high accuracy of 0.75 in clustering nursing notes, while other baseline methods, such as Expectation Maximization (EM), Hierarchical Clustering, and *k*-Means, have accuracy of 0.62 on average. A set of experiments on the benchmark 20-News Group dataset results in high accuracy of 0.8. We address the challenge of clustering within an imbalanced overlapping medical dataset using a semi-supervised learning framework.

7.1. Introduction

A growing number of seniors in the developed countries and their desire to live independently create the demand for automated health monitoring. In the last decade sensor networks emerged as a successful approach [6] [8] [10]. An automated health monitoring system captures daily activities using motion sensors and uses machine-learning methods to predict sign of illness based on the individual's history of activities [11] [12] [13]. It detects sensor sequences that represent abnormal activity, and annotates them with related medical terms. "Abnormal" days are defined as sensor activity patterns that require a nurse's assessment of the older adult and are documented in EHR.

Evaluating similarity between two time series, such as sensor sequences, has been of interest to researchers for decades [128] [129]. A number of distance functions to measure similarity between two sequences exist in the literature [130]. Defining each sensor of the sensor network as a separate dimension of the data produces a MATS. Then functions, such as DTW and LCSS, can measure the distance between two MATS [131] [132]. In chapter 3, our approach considers MATS as a one-dimensional sequence and employs TSW dynamic programming to measure similarity of two sensor sequences with respect to their time stamps [140]. We use this function to find sensor sequences most similar to the observed sensor sequence. The next step is to annotate them with appropriate medical terms from the medical notes.

Processing medical notes is still an open question in the domain of health informatics. Proper interpretation of the document is challenging because of complications, such as misspelled words, acronyms and numbers. Moreover, it needs domain expert knowledge to create clinically meaningful labels. These labels can be used in data mining applications, such as medical documents clustering. Labeling medical documents (such as nursing visit notes) is a cumbersome task, since it requires manual effort from the clinician. Semi-Supervised Learning is a promising solution to overcome this challenge [155] [156] [157].

In this paper, we propose a new framework for a health monitoring system that annotates abnormal sensor sequences with clusters of nursing visit notes. Our nursing

visit notes are an unbalanced dataset with overlapping clusters and few labeled data. In this paper, we aim to build an effective semi-supervised document clustering approach that automatically discovers informative constraints and finds a good document partition as well.

The first contribution of our approach is that we represent clusters by a language model. Most existing semi-supervised document clustering approaches are based on parametric models such that each cluster is represented by a set of parameters. For example, constrained *k*-means clustering algorithm is derived from the *k*-means document clustering algorithm [156]. In this approach, cluster centroids are considered as the parameters of the model and used to represent clusters. However, since documents usually follow multinomial distributions, these parametric models may not work well for the semi-supervised document clustering problem. The main reason is the distribution assumptions of the parametric model might not be valid assumptions for the underlying clusters. To address this problem, language modeling proposes a novel solution which has shown considerable success in information retrieval. This approach first represented by Ponte and Croft [158]. The basic idea behind language modeling is to use a non-parametric probability distribution as the document representation. In [159], authors used this approach and proposed an active learning framework based on language modeling.

The second contribution of our approach is to use bigrams model to take into account term co-occurrences in the language model. Generally, most of semi-supervised techniques use the ‘‘bag-of-words” model. In the bag-of-word approach, each text document is represented by a set of independent terms where each term can be a word or a token. However, terms might not be independent in reality. Specifically in health informatics domain, different terms can be used to capture similar ideas due to different sources and different writing styles of authors. Our approach relaxes this assumption by accounting term dependence relationships that can be captured by the bigrams model. To capture this dependency, we use user-provided pairwise constraints specifically for the terms related to the same cluster. The bi-grams model of terms in the documents labeled with the same cluster should have a higher probability of cohesiveness. Terms co-occurred frequently in the documents labeled with different clusters are relatively not so

discriminative. In [159], authors considered term-to-term co-occurrences to capture the term dependencies. However, this approach cannot perform well enough in medical notes such as nursing visit notes that usually are very short notes.

The third contribution of our work is that we address the challenge of unbalanced nature of dataset by our novel approach in defining the language model. Medical document are usually unbalanced because of different distributions of documents in various topics. Specifically, medical visit notes are highly unbalanced which challenges the document clustering task. Authors of [160] used a statistical approach to address this challenge. They proposed a novel algorithm based on analysis of variance (ANOVA), fuzzy C-means (FCM) and bacterial foraging optimization (BFO) to classify unbalanced data. ANOVA can measure the difference between the means of two or more groups in which the observed variance is partitioned into components due to various explanatory variables. ANOVA has the ability to select beneficial feature subsets. FCM has the ability to identify data into clusters with certain membership degrees, and BFO has the fast ability to converge to global optima. However, to get a desirable accuracy a large number of labeled training dataset is needed.

In another approach [161], authors used SVM to solve cluster unbalanced dataset. They investigated the feasibility and efficiency of SVM as a preprocessor. They analyzed different classification algorithms such as Multilayer Perceptron (MLP), Logistic Regression (LR), and Random Forest (RF). Various standard balancing techniques such as under-sampling, over-sampling and Synthetic Minority Over-sampling TEchnique (SMOTE) are also employed in this paper. The performance of this approach varies with different datasets.

In a novel approach, we use the MAP probability to address the problem unbalance clusters in dataset. In [162], Dirichlet priors are set on the parameters and authors used the EM algorithm to obtain MAP estimates of the parameters for an unsupervised text clustering. We take advantage of user-preferences as pairwise constraints to improve the prediction of MAP; hence, we enhance the performance of text clustering.

This chapter is organized as follows: in the next section we briefly review literatures for related work. In section 7.3, we describe the dataset we used for this study. In section 7.4, we describe our novel semi-supervised learning method to cluster medical documents. Section 7.5 discusses our experimental results. We finish this chapter in section 7.6 by discussion and conclusion.

7.1. Related Work

Document clustering algorithms are methods that can be used to partitions documents and can be divided into hierarchical document clustering methods and partitional document clustering methods [163] [164]. Hierarchical document clustering methods represent clusters by a hierarchical tree structure that are also known as a dendrogram. Mainly, there are two basic approaches in hierarchical document clustering techniques, namely, agglomerative document clustering approaches and divisive document clustering approaches. In agglomerative document clustering the hierarchical tree structure is discovered in a bottom up approach. Whereas, in divisive document clustering approaches clusters are discovered in a top down way.

Partitional document clustering methods group text documents into flat partitions with a fixed number of clusters. *k*-means method is the most widely used partitional document clustering approach. In [164], authors compared nine agglomerative algorithms and six partitional algorithms. They proposed a novel approach in agglomerative algorithms that uses clusters obtained by partitional algorithms to constrain the agglomeration process. Among different document clustering methods some of them are designed to deal with factors that improve the performance of clustering. Cai et al. [165] proposed a document clustering algorithm that handles the problem of high dimensional document space using locality preserving indexing. Li et al. [166] represented a document with a sequence of words. The number of shared frequent word sequences in two documents is used as a measure of their similarity.

Generally, user-provided information is not considered in the typical document clustering techniques to detect the user grouping preferences. However, semi-supervised document clustering techniques guide the document partitioning process with a small

amount of user-provided information. In this approach, user preferences are incorporated in the clustering process as constraints. Semi-supervised clustering methods are categorized based on the type of constraints, the stage where these constraints are applied into the clustering process, or the way constraints are used to improve the clustering method.

Constraints are divided into two categories: label of documents [156] [167] [168], and pairwise constraints (where two documents should assign to the same cluster or not) [169] [170] [171]. Moreover, there are some complex constraints reported in [172]. In this report, authors a more general type of constraints that indicates if a document should not be assigned to a specific cluster. Researchers also reported a type of complex constraints that specifies background information, and designed a triplet constraint that considers relative comparisons of documents [173] [174].

Another important factor in semi-supervised clustering with constraints is the stage where the constraints are applied to the clustering process. The user preferences can be capture off line and before the clustering process [175] [168] [176] [177]. In feedback-based methods users provide feedbacks in an iterative process in clustering algorithm [178]. Cohn et al. [172] proposed uses user preferences in the form of feedbacks to guide the iterations of the feedback-based semi-supervised clustering process. This approach let users choose documents for generating constraints.

Based on the way that user preferences are used semi-supervised document clustering methods are divided into three categories, constraint-based, distance-based, and a combination of distance and constraints. In constraints-based approach user preferences are directly used in the clustering process. The objective function of the clustering method is modified such that the user constraints are satisfied. Authors in [179] defined user constraints as must-link and cannot-link constraints and incorporate them into the clustering process [169]. In [180] [156], authors proposed two semi-supervised clustering based on *k*-means clustering which generates initial cluster seeds using user preferences and guides the clustering process.

Distance-based approach uses user preferences to learn a more accurate distortion measure over the data space and improve the performance of clustering [155] [181] [182]. The distortion measure is trained based on the constraints. In [175] [183], Xing introduced a new method to learn distance measure from similar data points. The idea is to pose metric learning as a convex optimization problem.

In a new approach [155] [184] Mahalanobis metric can be learnt using component analysis algorithm. The metric learning method performs nonlinear transformation globally and linear transformation locally. In [174] [185], Authors proposed a novel method that learns the underlying dissimilarity measure while finding compact clusters in the given dataset. Yan and Domeniconi [186] addressed the high dimensionality problem by projecting the data and constraints in multiple subspaces and learning positive semi-definite similarity matrices therein.

7.2. Dataset

For this chapter, we used a pilot dataset from six residents of TigerPlace as shown in Table 7-1. Motion sensors are placed in the living room, bedroom, bathroom and kitchen to capture various aspects of the residents' behavior. The reasoning system uses the activity data generated by sensor network to predict change in resident's health status. It processes a log file of the sensor events to extract the sensor sequences of "normal" and "abnormal" days in a predefined time interval. The EHR system stores nursing visit notes for each resident along with other medical information. If an abnormal day is detected, an alert manager notifies the clinician about potential problems. They can view data via a secure Web-interface and provide feedback in the EHR.

In this chapter, we propose a new framework that improves performance of the system for detecting abnormal events and generating alerts to the healthcare providers. Figure 7-1 shows our proposed pipeline for predicting abnormalities.

Table 7-1. Pilot dataset form sequence annotation.

Resident Code	Number of sensor days	Abnormal days	Nursing
1	560	200	235
2	615	210	220
3	580	195	200
4	610	215	225
5	600	205	240
6	620	200	235

A data logger collects sensor sequences and stores them in a sensor dataset. In a real time process, the detection engine searches the sensor dataset for past sequences similar to the currently observed one. The detection engine identifies abnormal events by employing the distribution of similar sensor sequences described in [23]. We assume that an abnormal sensor pattern is due to a medical condition, which is generally true for the elderly population of TigerPlace. If the observed sensor sequence is classified as "abnormal", a list of nursing notes from the days that include similar sequences is extracted from EHR and sent to the Alert Manager. The NLP engine uses a novel semi-supervised learning framework to label the identified event based on the cluster of selected nursing notes. Then the system sends the alert to the clinician with details about the possible abnormality. We describe our proposed semi-supervised learning framework for the NLP engine in the next section.

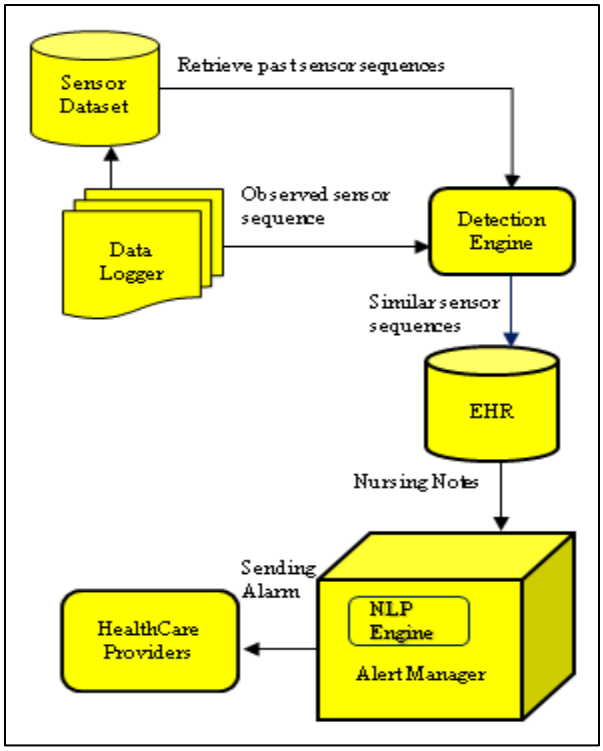


Figure 7-1. Sensor sequence annotation pipeline.

7.3. Clustering Nursing Notes

To detect the topic of the abnormality, we use nursing notes linked to previous illness episodes. In [187] we proposed a system that matched analogous abnormal events and reported shared medical terms as annotations. Unfortunately, the system was not efficient due to the complexity of nursing notes. In this paper, we propose to group nursing notes into clinically meaningful clusters using a semi-supervised approach with pairwise constraints, so that the NLP engine annotates the observed abnormal sensor sequence with the cluster label.

7.3.1. User judgment on document pairs: Pairwise constraints

We convert user-provided information about pairs of documents into pairwise constraints. "Must-link constraints" indicate the pairs of documents that should be assigned to the same cluster and "cannot-link constraints" specify pairs assigned to different clusters. Mathematically we denote the set of document pairs with must-link constraints as C_{ML} and the set of documents pairs with cannot-link constraints as C_{CNL}.

The set of all document pairs with user preferences is represented as $C = \{C_{ML} \cup C_{CNL}\}$. Now, we define each element of C_{ML} as pairwise must-link constraints:

$$P_{ML} = \{\Gamma(d_1, d_2) \,|(d_1, d_2) \in C_{ML}\} \quad (7.1)$$

Similarly, each element of C_{CNL} is denoted as pairwise cannot-link constraints:

$$P_{CNL} = \{\Gamma(d_1, d_2) \,|(d_1, d_2) \in C_{CNL}\} \quad (7.2)$$

Next, we formulate the concept of neighborhood used in the proposed semi-supervised clustering method.

7.3.2. Defining boundary of clusters: Neighborhood

User-provided constraints identify sets of documents that should be in the same cluster, or a "neighborhood" [19]. For example, documents with must-link constraints are grouped in the same neighborhood, whereas documents with cannot-link constraints are assigned to different neighborhoods. For two different neighborhoods, there must be at least one cannot-link constraint between them. For example, given the sets of C_{ML} and C_{CNL} as C_{ML} = {(d_1, d_2), (d_2, d_3), (d_5, d_6), (d_5, d_7)} and C_{CNL} = {(d_2, d_5), (d_7, d_1)}, two neighborhoods are shown in Figure 6-2. In this Figure, solid lines represent must-link constraints and dotted lines represent cannot-link constraints. Neighborhoods are outlined by dashed lines. Documents d_1, d_2, d_3 are in the same neighborhood, since they are in the C_{ML} set, whereas documents d_2 and d_5 are in a different neighborhood, since they are in the C_{CNL} set. Therefore, we can form two neighborhoods, n_1 and n_2. Next, we formulate the cluster definition.

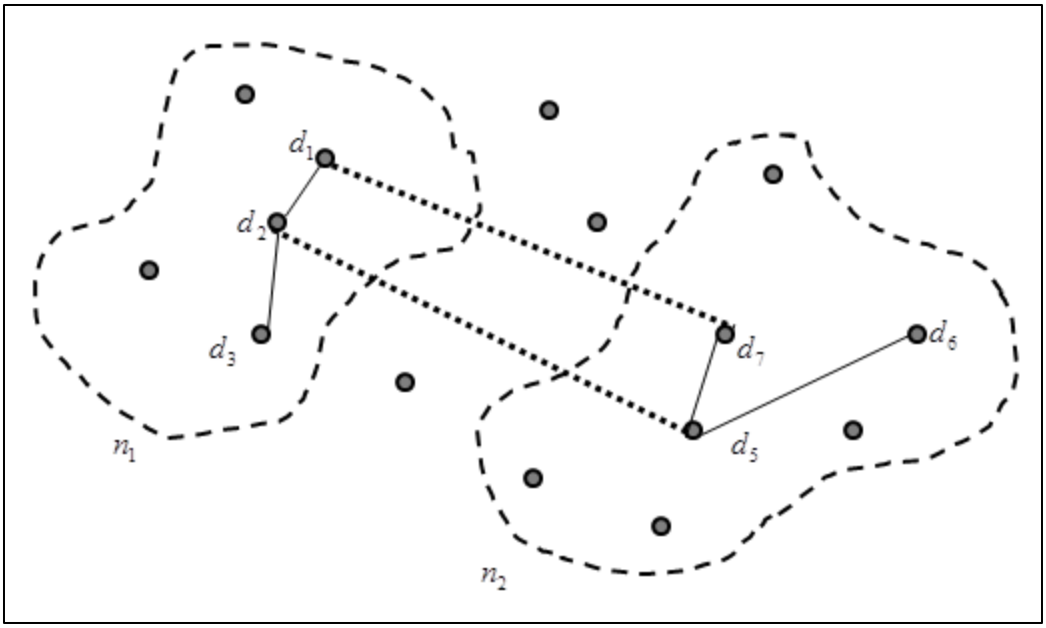


Figure 7-2. Neighborhood definition.

7.3.3. Representing clusters using term distribution

Backed by solid statistical theory, language modeling has been successfully applied to the problem of information retrieval [188]. This approach ranks documents using the likelihood of query with respect to an estimated language model. In our proposed framework we represent a language model for each cluster as M_c. Using a conventional language modeling approach, the cluster model is estimated based on the posterior probability of terms $p(t|M_c)$ [188]. Therefore, from Bayes' formula we have

$$p(M_c|t) \propto p(t|M_c)p(t), \tag{7.3}$$

where $p(t)$ is the prior belief that a term t is representative of the cluster model M_c, and $p(t|M_c)$ is the term likelihood, given the cluster model. Researchers have proposed different methods to estimate the term likelihood [189]. Guided by the Jelinek-Mercer smoothing method [190], we propose a new approach to estimate the term likelihood based on the term prior belief probability and terms maximum A-posterior probability. First we define the prior belief of terms.

7.3.3.1. Term's prior belief probability

We define the prior belief of terms based on a training dataset.

$$P_{pb}(t) = \frac{f_{t,train\_DS}}{|Training\ Dataset|} \quad (7.4)$$

The frequency of term t in the training dataset ($f_{t,train\_DS}$) normalized by the total number of terms in the training dataset (|*Training Dataset*|) gives an estimate of the prior belief of term, noted by $P_{pb}(t)$. Next, we define the term likelihood given the cluster model.

7.3.3.2. Terms Maximum A-Posterior Probability

There are different variants of term maximum likelihood estimation given the cluster model [188]. Jelinek-Mercer smoothing method defines maximum likelihood based on the probability of term occurrence in the training dataset. It assumes a uniform document distribution. However, this assumption is not valid for most medical document datasets, such as our nursing notes. Unbalanced nature of this data decreases performance of conventional document clustering methods. To address this challenge, we use MAP estimation of the term with a given cluster model. The MAP of term t given cluster model M_C is:

$$P_{MAP}(t|M_c) = \frac{\sum_{d\in D} P(M_c|d) f_{t,d} + \sum_{c=1}^{k} \sum_{d\in D} P(M_c|d) f_{t,d}}{\sum_{d\in D} P(M_c|d)|d| + \sum_{c=1}^{k} \sum_{d\in D} P(M_c|d)|d|} \quad (7.5)$$

where |*d*| is the number of terms in document *d*, $f_{t,d}$ is the frequency of the term *t* in document *d*, and $P(M_c|d)$ is the probability that document *d* should be assigned to the cluster model M_c. Next we formulate a cluster representation using the term prior belief and a MAP of terms.

7.3.3.3. Cluster Representation

We define a language model for each cluster. A cluster model is a linear interpolation of term prior belief and MAP of terms given the cluster model (equation 7.6).

$$P(t|M_c) = (1 - \lambda)P_{MAP}(t|M_c) + \lambda P_{pb}(t) \quad (7.6)$$

In equation 7.6, λ is the coefficient that controls the influence of each part of the cluster model. If λ=0, the cluster model is estimated solely using the MAP of the term. Whereas λ=1 only considers the prior belief of the term in the training dataset and discards the MAP of terms. Given the cluster model, the goal of the semi-supervised clustering is to form a set of clusters that maximizes the total probability of documents with respect to the user provided information that is pairwise constraints. In the next section, we describe this process in detail.

7.3.4. Semi-supervised clustering method

In our proposed semi-supervised clustering method, the goal is to partition documents into a set of clusters such that the probability of generating documents using user provided pairwise constraints is maximized. We assume that the documents are independently assigned to the clusters; therefore, the objective function of the language model can be expressed as a summation of all documents [167]. Since each single document has a set of terms, we calculate the probability of each document from its terms. We use a bi-grams model to express a document using its terms. In this model, we employ pairwise constraints to generate a bi-gram probability, as described in the following.

7.3.4.1. Terms bi-grams distribution

We use the pairwise constraints to estimate the bi-gram distribution. It improves the estimation of the relationship between bi-gram of terms within the same clusters. The probability of bi-gram of terms in must-link constraints should be much higher than the probability of bi-gram in cannot-link constraints. The probability of bi-gram of terms t_1

and t_2 is denoted by$P_\psi(t_1, t_2)$. We use both must-link and cannot-link constraints to estimate $P_\psi(t_1, t_2)$, so the estimated bi-gram of terms t_1 and t_2 from must-link constraints is denoted by $P_{ML}(t_1, t_2)$ and from cannot-link constraints by $P_{CNL}(t_1, t_2)$. Both are calculated using the bi-gram frequency counts of terms [191]:

$$P_{ML}(t_1, t_2) = \frac{count_{ML}(t_1,t_2)}{f_{ML}(t_1)} \tag{7.7}$$

$$P_{CNL}(t_1, t_2) = \frac{count_{CNL}(t_1,t_2)}{f_{CNL}(t_1)} \tag{7.8}$$

where $count_{ML}(t_1, t_2)$ is the frequency of bi-grams (t_1, and t_2) in the must-link documents, $count_{CNL}(t_1, t_2)$ is the frequency of bi-grams (t_1, and t_2) in the cannot-link documents, $f_{ML}(t_1)$ is the frequency of t_1 in must-link constraints, and $f_{CNL}(t_1)$ is the frequency of t_1 in cannot-link constraints. $P_\psi(t_1, t_2)$ is a linear interpolation of $P_{ML}(t_1, t_2)$ and $P_{CNL}(t_1, t_2)$.

$$P_\psi(t_1, t_2) = \gamma P_{ML}(t_1, t_2) + (1 - \gamma)P_{CNL}(t_1, t_2) \tag{7.9}$$

where γ is a smoothing parameter in the range of 0 and 1 that controls the influence of each set of pairwise constraints on $P_\psi(t_1, t_2)$.

7.3.4.2. Objective function

We use the objective function described in [178]. It is formulated as:

$$P(d|M_c) = \prod_{t \epsilon d}\{\beta \sum_{t_2 \neq t} P_\psi(t, t_2)P(t_2|M_c) + (1 - \beta)P(t|M_c)\} \tag{7.10}$$

$$\mathcal{J}_{\text{obj}} = P(D) = \prod_{d \epsilon D} P(d) = \prod_{d \epsilon D} \sum_{c=1}^{k} \pi_c P(d|M_c) \tag{7.11}$$

where d is the document, c is a cluster, k is the total number of clusters, t_1 and t_2 are distinct terms, β is a smoothing parameter in the range of 0 and 1 that controls the influence of bi-gram distribution to the clustering process, and π_c is the prior probability of the cluster c. The objective function is maximized through a two-step iterative process. In the first step, the probability of assigning document d to the cluster c in the iteration i, $P^i(M_c|d)$ is estimated based on this probability in the previous step as follows [167]:

$$P^i(M_c|d) = \frac{\pi_c^{i-1} P^{i-1}(d|M_c)}{\sum_{c'=1}^{k} \pi_{c'}^{i-1} P^{i-1}(d|M_{c'})} \quad (7.12)$$

In the second step, we use $P^i(M_c|d)$ from the first step to estimate the prior probability, π_c^i as follows:

$$\pi_c^i = \frac{\sum_{d \epsilon D} P^i(M_c|d)}{|D|} \quad (7.13)$$

where $|D|$ is the total number of documents in the training dataset *D*. Further, we evaluate the performance of the proposed method. The proposed semi-supervised document clustering algorithm is presented in Figure 7-3.

Input: set of data points$\mathcal{X} = \{x_i\}_{i=1}^N$, number of clusters K, a set of must-link constraints ML, a set of cannot-link constraints CL.

Output: K clusters of X such that objective function $\mathcal{J}_{obj}$ (Eq. 6.11) is minimized.

Algorithm:

1. Initialize the K clusters centroids $\{\pi_c^{(0)}\}_{c=1}^K$, set i=0.
2. Repeat until convergence
    a. Given $\{\pi_c^{(i)}\}_{c=1}^K$ re-assign cluster labels $P(M_c|d)^i$ on the points $\{x_d\}_{d=1}^N$ to minimize $\mathcal{J}_{obj}$ (Eq. 6.11).
    b. Given cluster labels $P(M_c|d)^{i+1}$, re-calculate cluster centroids $\{\pi_c^{(i+1)}\}_{c=1}^K$to minimize $\mathcal{J}_{obj}$ (Eq. 6.11).
    c. Update $\mathcal{J}_{obj}$
    d. i= i+1

Figure 7-3. Semi-supervised document clustering pseudo code.

7.4. Experimental Results

In this section, we demonstrate performance of our proposed method on two datasets: the TigerPlace dataset and a benchmark 20-News group dataset.

7.4.1. TigerPlace Nursing Notes dataset

We use the proposed clustering method to extract nursing visit notes from the TigerPlace EHR system (see Figure 7-1). Then we use the cluster labels to annotate abnormal sensor sequences. Below we briefly describe the nursing notes dataset and demonstrate the preprocessing.

7.4.1.1. Nursing Visit Notes Dataset

We use nursing visit notes from 6 TigerPlace residents. Our medical experts analyzed a small subset of nursing notes and manually labeled five clusters for each resident. Since clusters differed between residents, we trained our system to personalize them for each person. Table 7-2 shows the number of nursing notes for each resident, with and without labels, the number of terms, and the number of visit notes in each category (label).

Table 7-2. TigerPlace nursing visit notes dataset.

Resident Code	Nursing visit notes	Number of terms	Labeled visit notes	Unlabeled visit notes	Labels				
1	235	450	50	185	Fall	Plan of Care	Acute Events	Medication	Cognitive Change
					11	21	7	21	14
2	220	426	50	170	Follow up	Plan of Care	Acute Events	Medication	Pain
					12	17	9	14	5
3	200	385	40	150	Assessment	Plan of Care	Symptoms	Medication	Wound Dressing
					10	16	8	15	7
4	225	410	50	175	Assessment	Plan of Care	Symptoms	Medication	Psychosocial
					20	18	12	9	14
5	240	450	45	195	Assessment	Plan of Care	Symptoms	Medication	Acute Events
					21	23	10	25	6
6	235	432	50	185	Assessment	Plan of Care	Symptoms	Medication	Acute Events
					17	20	12	22	8

7.4.1.2. Preprocessing

In our experiments, we represent each nursing visit note by a feature vector created through a preprocessing mechanism. We extract nursing visit notes from the EHR system in a plain text format. Then, we use cTAKES to map words to CUI from UMLS. cTAKES is an open source Apache software consisting of java libraries [192]. We use this tool in offline and batch mode to represent each nursing note as a vector of CUI and its frequency. We use this vector in the training and testing phases.

7.4.1.3. Evaluation metrics

We evaluate performance of the proposed semi-supervised method by a commonly used metric, a pairwise F-measure. It is frequently used to assess performance of semi-supervised learning methods with constraints. This measure is calculated as a combination of Recall (R) and Precision (P), as follows:

$$P = {}^{q_c}/_{q_s} \tag{7.14}$$

$$R = {}^{q_c}/_{q_f} \tag{7.15}$$

$$F - measure = \frac{2.P.R}{P+R}, \tag{7.16}$$

where q_c is the number of nursing notes pairs in must-link constraints that are correctly assigned in the same cluster; q_s is the number of notes pairs that are assigned to the same cluster; q_f is the number of notes pairs that are actually in the same cluster [157][18].

7.4.1.4. Semi-supervised clustering performance on TigerPlace dataset

To set up the experiments, we initialize three parameters in our system. The first parameter is λ in Eq.(7.6). It controls the influence of MAP of terms in the cluster model. We test different values for this parameter, as shown in Figure 7-4(a). The results indicate that λ=0.3 maximizes the pairwise F-measure. The second parameter is γ in Eq.(7.9). It controls the contribution of must-link versus cannot-link constraints on the bi-gram distribution. We show the results of different values of γ in Figure 7-4(b). Results indicate $\gamma = 0.6$ maximizes the pairwise F-measure. The third parameter is β in

Eq.(7.10). The result of different initial values of this parameter is plotted in Figure 7-4(c). We set β = 0.4. The bi-gram distribution and occurrence of terms in the cluster affect performance of the semi-supervised clustering.

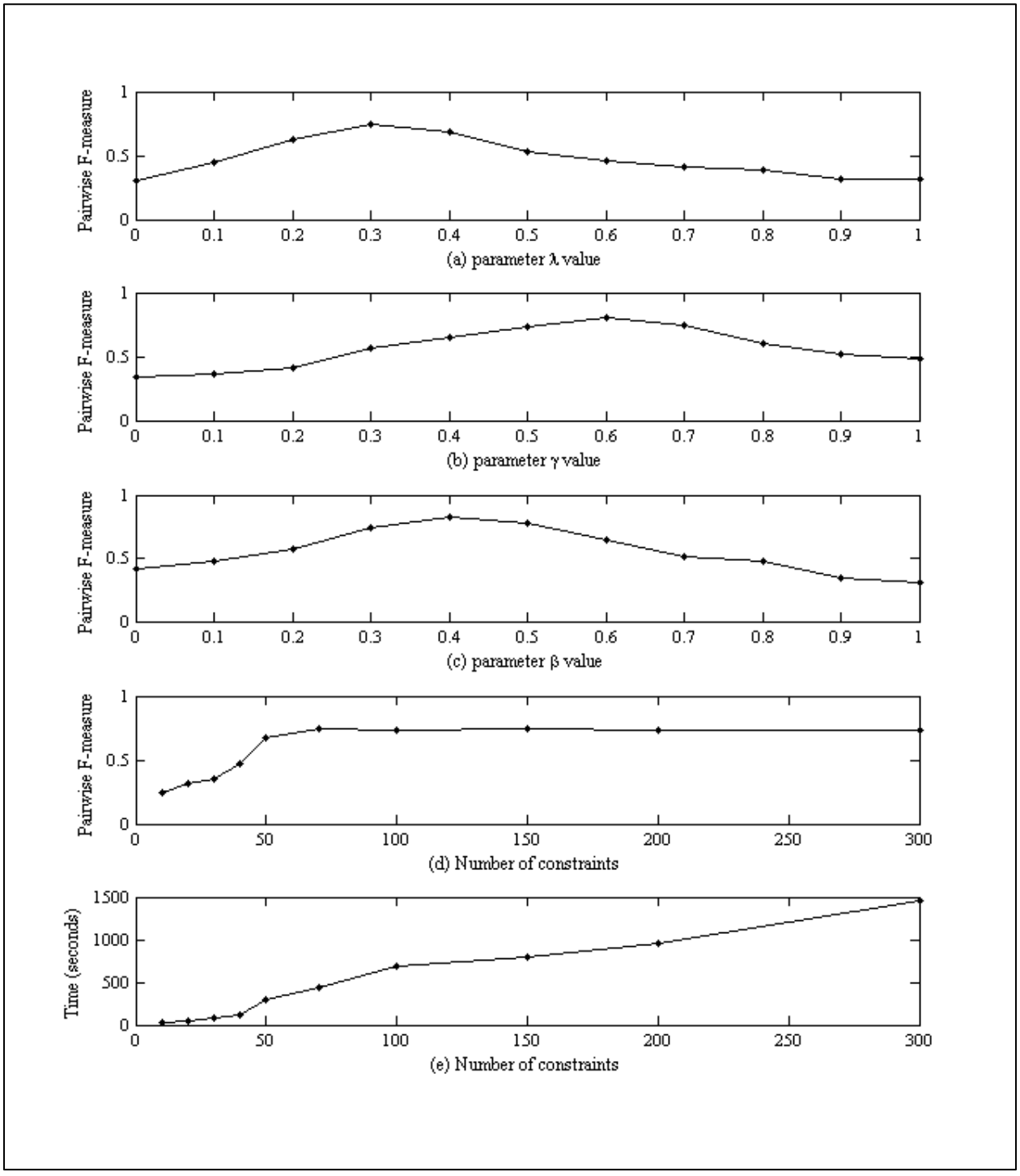


Figure 7-4. Parameter initialization of semi-supervised clustering on TigerPlace dataset.

We also evaluated the effect of the number of constraints on the performance of the proposed semi-supervised clustering method, as shown in Figure 7-4(d). For this experiment, we simulated user-provided decisions as pairwise constraints. If two nursing visit notes are assigned with the same cluster label, then we consider them as a must-link constraint. If there are no common labels on two nursing visit notes, then we consider this pair as a cannot-link constraint. Increasing the number of constraints enhances the performance of the proposed semi-supervised clustering method from 0.24 F-measure with 10 pairwise constraints to 0.75 F-measure with 75 pairwise constraints. Figure 7-4(e) depicts the execution time in seconds for different number of constraints. Raising the number of pairwise constraints increases the execution time. The proposed semi-supervised clustering method performs very well on the nursing notes dataset. Using 75 pairwise constraints, the proposed method detects all clusters (5 clusters) with 0.75 F-measure in retrieving the test dataset labels. We compared our method with baseline methods, such as Expectation Maximization, Hierarchical Clustering, and *k*-means clustering. They result 0.58, 0.61, 0.65 of F-measure respectively.

7.4.1.5. An example of health pattern detection

In this section, we illustrate the process of health pattern detection in Figure 7-5.

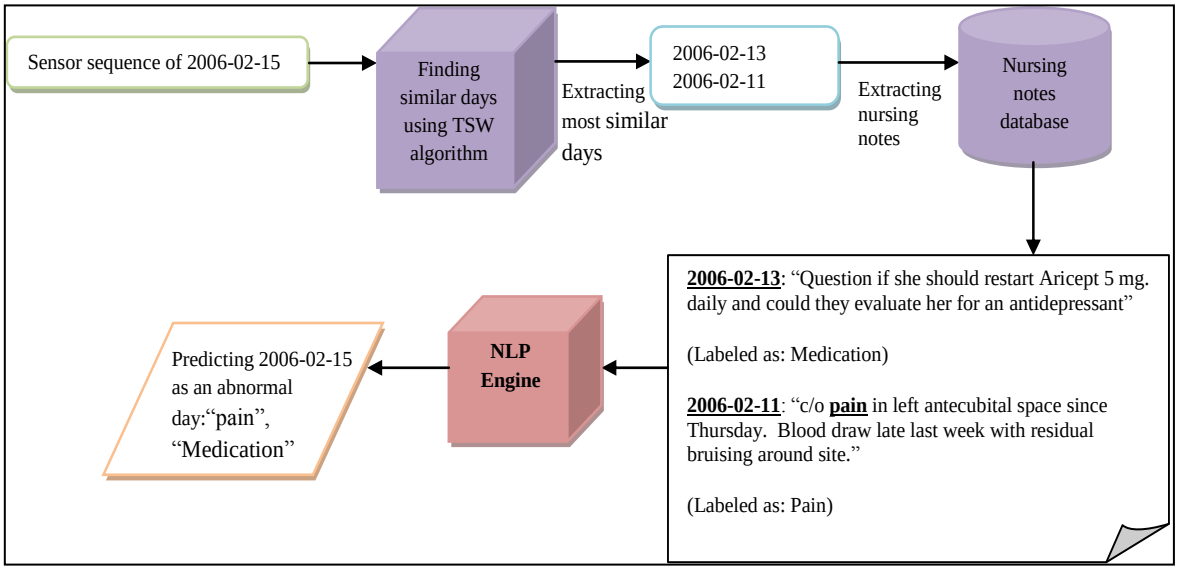


Figure 7-5. Example of predicting abnormal days using nursing notes.

For example, given an observed sensor sequence from resident #2 on February 15, 2006, we want to classify it as "normal" or "abnormal" and annotate the sensor sequence

of an abnormal day with related medical topics. The TSW algorithm finds sensor sequences in the stored database for this resident most similar to the given pattern (that is the sequence of given day). If similar sensor sequences are identified as abnormal, the system extracts associated nursing notes. The NLP engine finds clusters which those nursing visit notes belong to and annotates the observed sensor sequence with those labels. In Figure 7-5, the system identifies the sensor sequence of 2006-02-15 as an abnormal day because similar days, such as 2006-02-13 and 2006-02-11, are tagged as "abnormal". It also proposes that issues concern "pain" and "medication".

7.4.2. 20-News Group dataset

In the following section, we test our proposed method on a benchmark dataset. The "20-Newsgroup" is a collection of approximately 20,000 news articles, originally collected by Ken Kang in 1995 [193][30] and partitioned into 6 super groups and 20 subgroups. This dataset is a popular benchmark for evaluating text mining techniques, such as text clustering. We use a subset of this data for our experiments, shown in Table 7-3.

Table 7-3. 20-Newsgroup dataset.

Number of documents	**Number of terms**	**Labeled document**	**Unlabeled documents**	**Labels**				
600	3860	100	500	Computer. hardware. ibm	Computer. hardware. mac	Sport. hockey	Science. medicin	Science. electronics
				20	35	25	30	25

7.4.2.1. Preprocessing

This dataset is divided into 3 super topics: "computer", "science", and "sport". The Computer group has two sub groups, "hardware IBM" and "hardware MAC". The Science group also has two subgroups, "medicine" and "electronics". Subgroups of one super class are highly related to each other, whereas subgroups from different super classes are highly unrelated. Each document is in the plain text format. To extract the

feature vector of each document, we use the "Word Vector Tool" [194], a collection of Java libraries designed for statistical language modeling. We create the word vector representations of text documents in the vector space model. In this model, a document is represented by a vector that denotes the relevance of a given set of terms for this document [194]. We first remove stop words using a Standard English stop word list provided in the tool. Then we normalize the text with "Snowball Stemmer", a tool that maps different grammatical forms of a word to the common term [194]. We generate the document vector as a set of terms and its frequency pairs based on the following notations:

f_{ij} : the number of occurences of term i in the document j (7.17)

fd_j : the total number of terms occuring in document j (7.18)

ft_i: the total number of documents in which term i appears at least once (7.19)

TF_IDF: the tf_idf measure with $v_{ij} = \frac{f_{ij}}{fd_j} log\left(\frac{|D|}{ft_i}\right)$ (7.20)

where $|D|$ is the total number of documents. The "Term Frequency" is the relative frequency of a term in a document. Finally, using this preprocessing pipeline, we convert each news document into a feature vector that will be used for experiments on the Semi-Supervised method. We used the same evaluation metric as described in section 7.4.1.3.

7.4.2.2. Semi-supervised clustering performance on 20-Newsgroup dataset

Similar to the TigerPlace dataset, we run a set of experiments to initialize the parameters λ, γ, and β. Figure 7-6 plots (a), (b), and (c) show the results for each parameter with respect to the pairwise-F-measure. We chose λ = 0.4, γ = 0.5, and β = 0.6 to achieve the best performance. Figure 7-6(d) shows the effect of the number of constraints on the pairwise F-measure. We used the same simulation as described in section 7.4.1.4 to generate user provided judgments (pairwise constraints). Increasing the number of constraints enhances the performance of our semi-supervised clustering method, from 0.32 F-measure with 50 pairwise constraints to 0.8 F-measure with 500 pairwise constraints. Figure 7-6(e) shows that the execution time increases rapidly by

adding more pairwise constraints. Our proposed semi-supervised clustering method has a high performance on the 20-Newgroup dataset as well. Using 500 pairwise constraints, the proposed method detects all clusters (5 clusters) with 0.8 F-measure in retrieving the test dataset labels.

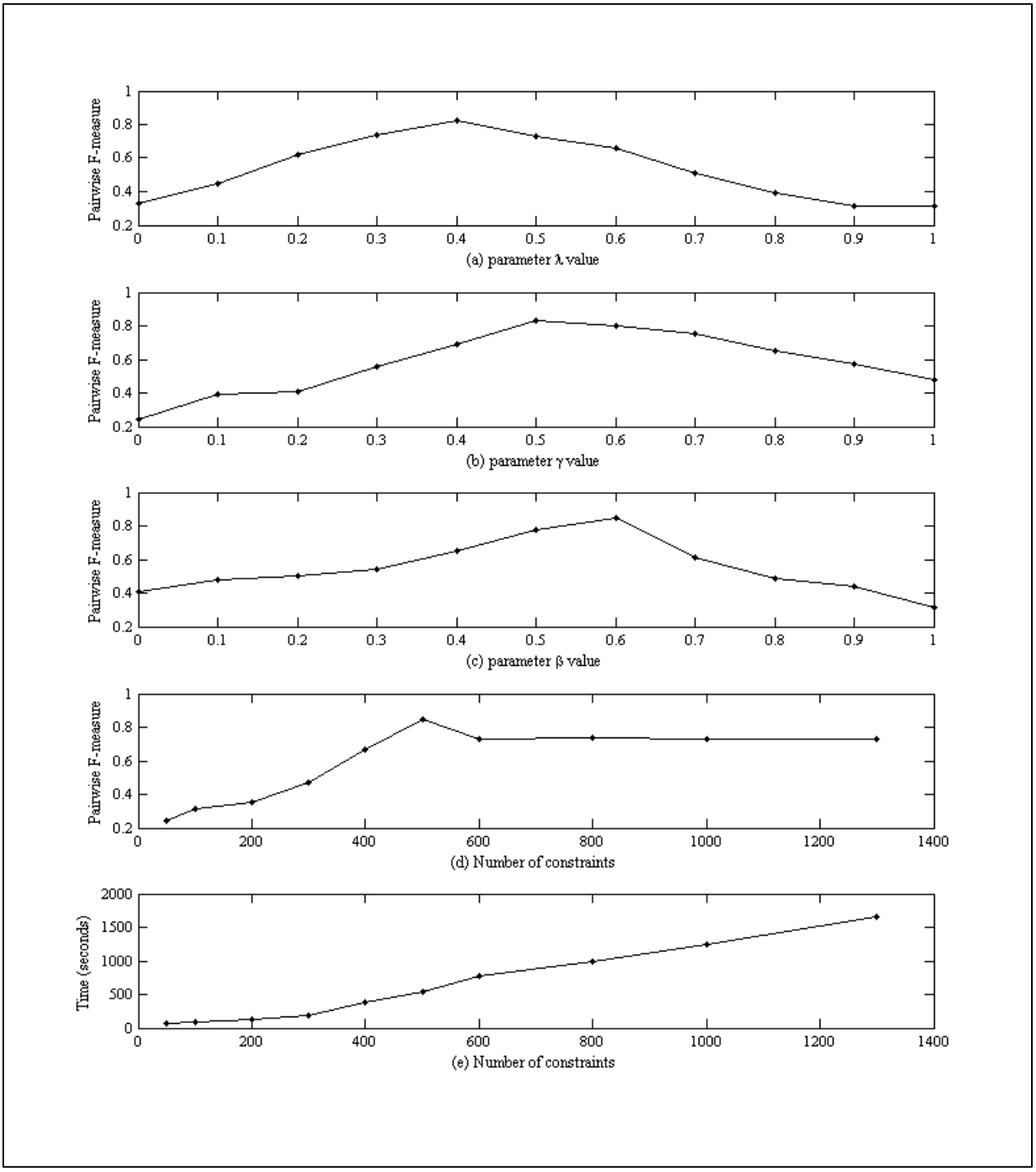


Figure 7-6. Parameter initialization of semi-supervised clustering on 20-Newsgroup dataset.

7.5. Conclusion

In this chapter, we present a novel framework for detecting early signs of illness in an automated health monitoring system for seniors. Our semi-supervised learning method labels health patterns, detected by a sensor network, with clusters of NLM CUI terms extracted from the nursing notes in our TigerPlace EHR dataset. It is designed to handle overlapping, unbalanced medical document data. We tested our framework on the TigerPlace dataset and a benchmark "20-Newsgroup" dataset. It outperformed baseline methods such as EM, Hierarchical Clustering, and *k*-means clustering. The goal of our framework is to improve the alert manager that notifies clinicians of health changes and allows for early intervention.

Chapter 8 Evaluation of Two Early Illness Alerts Methodologies in TigerPlace

TigerPlace uses sensor networks technology to bring an independent, smart home environment for its elderly residents while providing specific aids such as housekeeping, meals, and other services. Since 2010, smart home technology setting at TigerPlace assisted care providers in understanding behavioral patterns or early signs of illnesses that could be helpful in early illness interventions. The smart home setting provides a framework for using passive in-home sensor networks to collect sensor data. It uses Early Illness Alert (EIA) algorithms to model and detect signs of early illnesses, sends single-dimensional alerts that notify nursing care-providers, and collects clinical feedbacks on alerts from a team of clinical researchers with an expertise in gerontology.

The collected feedbacks provide valuable ground truth that is utilized to analyze and improve EIA algorithms. In this chapter, we compare our EIA algorithms described in chapters 4,5, and 6 to the method that have been used at TigerPlace since 2010 to detect abnormal events and generate EIAs.

8.1. Introduction

The main idea of an EIA algorithm is to detect if patterns or changes in the sensor data could have indicated the onset of a critical health event. The results of several case studies [196] show that the sensor data changed noticeably before a resident's hospitalization. The key concept of EIA algorithms is to detect this change in the sensor data. In the current EIA system at TigerPlace the initial idea is to take a continuous sliding window of two weeks of sensor data and build a "normal" model for each of the initial alert parameters, and then determine when any of the sensor signals deviate significantly from their normal values.

In the rest of this chapter, we describe the EIA algorithm that is currently being used at TigerPlace in more details. We discuss the parameters settings of this method and our training dataset that will be presented in section 8.2.6, table 8-4. We evaluate the performance of the TigerPlace EIA algorithm on the labeled dataset from TigerPlace. We finish this chapter by discussing the results of a comparison between the performance of the current TigerPlace EIA algorithm and our proposed method using TSW similarity measure.

8.2. Single-Dimensional Early Illness Alert Algorithm

The TigerPlace EIA method performs on sensor data on a daily bases. This method aggregates sensor data and assumes a Normal distribution of sensor hits. Following we discuss data representation and alert generation in more details.

8.2.1. Data representation

TigerPlace EIA method uses aggregated motion sensor data. It considers sensor data of each day separately and shows changes in the motion sensor activities of residents in hourly intervals. Activities are presented in Table 8.1. The hourly (1-24) aggregated sensor values are calculated for each activity based on the motion sensor log files. Each

day is represented by a 4*24 matrix (each row for an activity and each column for aggregated motion hits in an hour).

Table 8-1. Activities and related sensors

Activity	**Sensors**
Bedroom activity	Bed motion, Pulse, Breathing, Restlessness, Bed movement
Bathroom activity	Closet, Laundry, Shower, Bathroom motion
Kitchen activity	Kitchen motion, cup cabinet, refrigerator, stove, High temp, Low temp, cabinet, Drawer, Plate Cabinet, Silverware Drawer,
Living Room activity	Living room motion, on chair , off chair, Dining Room motion,

8.2.2. Error! Reference source not found.

The TigerPlace EIA method operates as follows: Given a sliding window of two weeks, calculate the alert parameter values for a given alert period for each day in the sliding window. With enough valid alert parameter values in the sliding window, calculate the sample mean and sample standard deviation of the valid values to model "normal" for a given alert parameter at that day. If the alert parameter value calculated for the day after the sliding window deviates (by 4 std) from the sample mean of the sliding window by enough sample standard deviations of the window, generate either an increase or decrease EIA. Figure 8-1 shows the pseudo code of this method.

Algorithm: Single-Dimensional Early Illness Alert

BEGIN

Given the sliding window length in days, L
Given alert parameter function $\rho(t, P)$, where t is a date and P is an alert period
Given alert period, $P \in \{Full\ Day\}$
Given valid alert parameter value threshold, T
Alert parameter values, $\boldsymbol{\rho} = \{ \rho_1, \rho_2, \ldots, \rho_L\} = \{0, 0, \ldots, 0\}$
FOR (n = 1 to L)
**Calculate $\rho_n = \rho(t_n, P)$
END FOR
$\boldsymbol{V} = \{v \in \boldsymbol{\rho} \mid v > T\}$
$\mu = mean(\boldsymbol{V})$
$\sigma = std(\boldsymbol{V})$
CASE $(\rho(t_{n+1}, P) - \mu \geq \sigma \cdot \sigma_{Increase})$
Send Increase EIA
CASE $(\mu - \rho(t_{n+1}, P) \geq \sigma \cdot \sigma_{Decrease})$
Send Decrease EIA
END
** The alert parameter functions for each alert parameter are defined in table 3.5
$\sigma_{Increase}$: Deviation threshold for Increase EIA
$\sigma_{Decrease}$: Deviation threshold for Decrease EIA

Figure 8-1. TigerPlace EIA algorithm

The original EIAs are single-dimensional because each alert represents either an increase or a decrease in a single alert parameter (activity) and for a single alert period. There are currently 4 alert parameters and 2 different alert types (increase and decrease). The alert parameters and how they are calculated on a daily basis are listed in Table 8-2.

Table 8-2. Alert parameter modeling.

Alert Parameter	Calculation of the Parameter Value $\rho(t, P)$, given a day t and period P
Bathroom Activity	No. of sensor events from bathroom motion, shower, etc.
Bedroom Activity	No. of weighted sensor events from bedroom motion and bed sensors
Kitchen Activity	No. of kitchen motion sensor (kitchen, fridge, etc.) events
Living Room Activity	No. of living room sensor events

The initial alert parameters consists of combining sensors in similar areas of the apartment, aggregating the bed restlessness in a manner which puts higher weights on higher levels of restlessness, and keeping the multiple levels of bed breathing and bed pulse separate in hourly intervals. Equation (8.1) formulates the bedroom activity.

$$BedroomActivity = \sum_{i=1}^{4} i * Restlessness_i + \sum_{j=1}^{3}(Pulse_j + Breathing_j) \quad (8.1)$$

8.2.3. Sliding Baseline

The EIA algorithm needs a set of training data to calculate the parameters of normal distribution such as mean and standard deviation. One way to do so is to pick a static range of time to calculate the mean and standard deviation for each periodic behavior for each elderly resident. This is less practical, because a suitable range of time must be hand-picked and may be tedious to find. Moreover, the range of time selected may represent a "normal" behavior pattern that is different from the resident's current normal, i.e. a resident's "normal" changes over time. As a practical solution in terms of implementation and to account for slower changes in a resident's normal behavior, the early illness algorithm uses a sliding baseline. We discussed the sliding baseline size with our clinical researchers and done several exploratory experiments. As a result, for TigerPlace motion sensor data the sliding baseline spans a time frame of two weeks, or 14 days (see Figure 8-2).

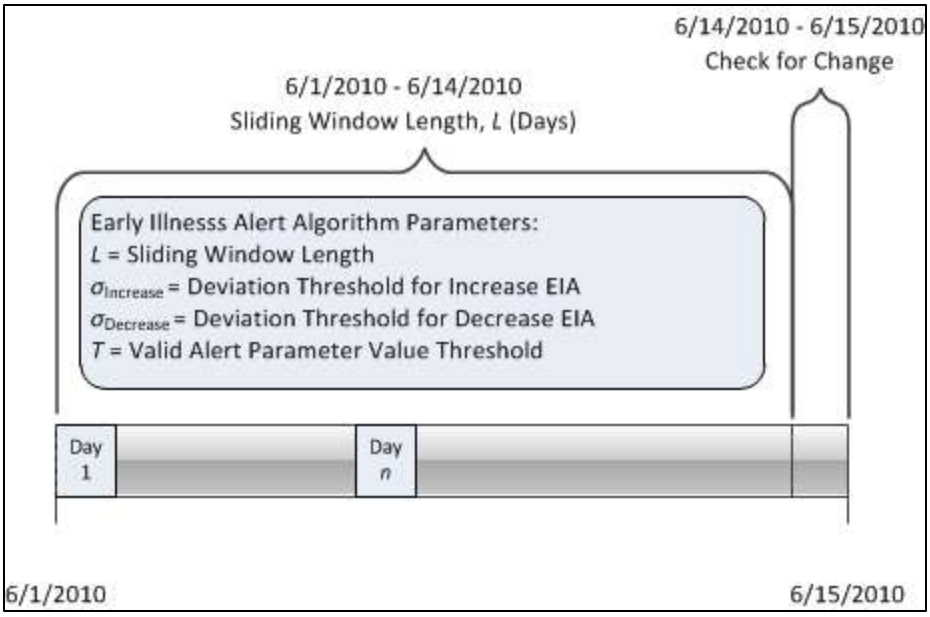


Figure 8-2. Early Illness Algorithm Sliding Window

8.2.4. Standard Deviation Threshold

With a constant sliding baseline size, the standard deviation-based threshold has a direct effect on the number of EIAs that are generated. As a result, the clinical researchers require a standard deviation threshold that is low enough to generate early illness alerts in situations where they are desirable but high enough so that they are not overwhelmed with false alarms. A retrospective study analyzed the effect of the standard deviation threshold for residents during a time period where early illness alerts would have proved useful [197]. The number of alerts as well as the time of alerts vs. the standard deviation threshold was analyzed, and the clinicians decided on a threshold that minimized the false alarms while maintaining the desired early illness alerts. A threshold of 4 standard deviations was chosen ($\sigma_{increase} = \sigma_{decrease} = 4std$).

8.2.5 Filtering Malfunctioning Sensors and Extended Absence

One important problem with real world sensor data is missing data due to battery failure, falling out of place, or sensor malfunction. Other possibilities are sensor dependent. For example, sometimes the pneumatic strip of the bed sensor is misplaced, or the resident sleeps in an awkward position that is not directly on top of the transducer. Other times, the staff or the resident removes the bed sensor without notifying the ElderTech team.

To filter out these events, a valid parameter value threshold T filters out periodic behavior values on days below that threshold from the sliding window. This filter may result in a training set that is less than the L days in the sliding baseline. If the sensor parameter value for the new day is less than T, then the algorithm will not generate an EIA. T was initially set to 0 to simply eliminate days where the periodic behaviors are absent. However, with the presence of TigerPlace staff there is a possibility of setting off a few sensor events while the resident is away. Thus, T was increased to 10.

8.2.6. Experimental results

This section presents the experimental results from TigerPlace EIA method.

8.2.6.1 Dataset

We gathered motion sensor data from 8 TigerPlace residents from November 1st of 2010 to July 13th of 2011. Clinical researchers at TigerPlace manually evaluated alerts and assigned rates for them. Table 8-3 shows five levels for alerts ratings. Table 8-4 shows the number of days with one or more alerts.

Table 8-3. Alert Feedback Ratings

Rating	Description
1	Clinically irrelevant
2	Less clinically relevant
3	Somewhat clinically relevant
4	Clinically relevant
5	Very clinically relevant

Table 8-4. TigerPlace dataset

Resident #	Number of days with alarms
1	42
2	33
3	31
4	38
5	44
6	86
7	43
8	54

After having discussions with our clinical researchers, we decided to rate alerts based on their ratings. An alert with a rating greater than or equal to 3 is considered a "GoodAlert", otherwise it is a "PoorAlert". Consequently, we labeled days as "abnormal", if they contained at least a "GoodAlert" and as "normal" otherwise. Table 8-5 summarizes the number of alerts for each activity and each resident using TigerPlace EIA algorithm.

Table 8-5. Labeled alerts using TigerPlace EIA method

Resident#	Activity	PoorAlert	GoodAlert	Total
#1	Bathroom	9	12	21
	Kitchen	8	6	14
	LivingRoom	13	11	24
	Bedroom	17	38	65
	Total Alerts	47	67	124
#2				
	Bathroom	3	13	16
	Kitchen	6	6	12
	LivingRoom	1	3	4
	Bedroom	12	37	49
	Total Alerts	22	59	81
#3				
	Bathroom	11	7	18
	Kitchen	13	5	18
	LivingRoom	22	9	31
	Bedroom	0	0	0
	Total Alerts	46	21	67
#4				
	Bathroom	5	6	11
	Kitchen	3	2	5
	LivingRoom	5	12	17
	Bedroom	4	10	14
	Total Alerts	17	30	47
#5				
	Bathroom	8	9	17
	Bedroom	6	13	19
	Kitchen	4	2	6
	LivingRoom	4	0	4
	Total Alerts	22	24	46
#6				
	Bathroom	5	1	6
	Kitchen	17	3	20
	LivingRoom	13	1	14
	Total Alerts	35	5	40
#7				
	Bathroom	4	2	6
	Bedroom	22	35	57
	Kitchen	7	1	8
	LivingRoom	8	4	12
	Total Alerts	41	42	83

#8				
	Bathroom	7	42	49
	Bedroom	11	94	105
	Kitchen	13	23	36
	LivingRoom	11	27	38
	Total Alerts	42	186	228

With the ability to continually and conveniently collect clinical feedbacks via automatically generated EIAs, the alert database becomes a valuable tool for further analysis of the EIA algorithm. With quantitative ratings available, ground truth can be created to separate the alerts the nursing clinicians view as having clinical relevance and merit an intervention, and those they view as false alarms, or simply good versus poor alerts.

8.2.6.2 EIA Results

This section presents the accuracy of the TigerPlace EIA method averaged over 8 residents of TigerPlace. Figure 8-3 shows the ROC curves of each alert parameter (activity) and Table 8-6 the area under the curves (AUC).

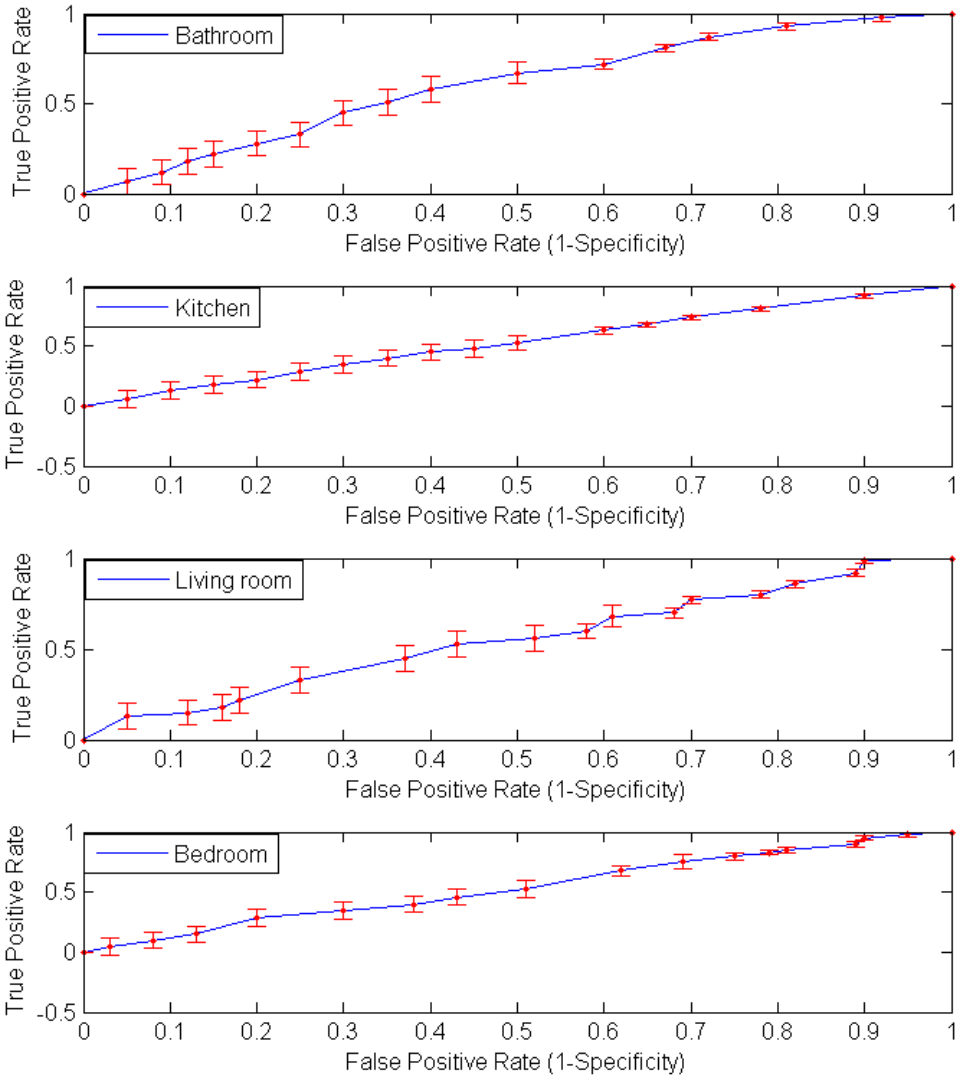


Figure 8-3. TigerPlace EIA ROCs on TigerPlace dataset. The plots show the average results on all TigerPlace residents. Error bars are shown in red color indicating variations of results.

Table 8-6. TigerPlace EIA algorithm average area under ROC curve

Activity	Area Under ROC curve
Bathroom	0.57
Bedroom	0.57
Kitchen	0.54
Living room	0.53

Interestingly enough, the bathroom activity EIAs at night did not discriminate between normal and abnormal days even though the clinical researchers mentioned numerous times that bathroom EIAs at night tended to be more clinically relevant. For the kitchen activity and living room activity parameter ROC curves, there were very few good EIAs in those two categories compared to the total number of EIAs, and most of the EIAs for those two categories at night were rated poor. The bathroom activity and bed restlessness EIAs had a more even distribution of good and poor alerts. This simply indicates the fact that a single EIA feature based on normal distribution cannot account for user activities. In the next section, we present the result of TSW measure on EIA in order to account for more of the possible clinically relevant situations.

8.3. Early Illness Alert Using TSW Similarity Distribution Approach

In this section, we use the same technique we discussed in chapter 5 for EIA. We compare the results of this method with single-dimensional EIA method.

8.3.1 TSW-EIA Method

Given n normal sensor sequences $\{S_i\}_{i=1,n}$ (for n days) we compute the pair-wise similarities between them, $\{s_{ij}\}_{i,j=1,n}$ using TSW. We, then, calculate the distribution of the $\{s_{ij}\}$ similarities of these "normal" days assuming they follow a Gamma distribution. Assume that we found out that $\{s_{ij}\}$ follows a distribution Gamma with parameters a and b, i.e. $\Gamma(a,b)$. To classify an unknown sequence S_x, we start by computing its similarities

with all normal sequences $S_{1,}\ldots S_n$, obtaining similarities $\{s_{ix}\}_{i=1,n}$. Then, we find the maximum of $\{s_{ix}\}$, $s_{x,max}$. The confidence that S_x is abnormal, $C(S_x)$ is 0 if $s_{x,max}>mean(\Gamma(a,b))$, otherwise equal to $1\text{-}P$ where P is the likelihood that $s_{x,max}$ comes from $\Gamma(a,b)$ and is calculated using:

$$P(s_{x,max}, a, b) = \int_0^{s_{x,max}} \left(\frac{1}{\Gamma(a)b}\left(\frac{t}{b}\right)^{a-1} e^{-\frac{t}{b}}\right) dt \quad (8.2)$$

The method is described in more details in Chapter 5.

8.3.2 Experimental Results

For experiments, we used rated alerts to label days as "normal" and "abnormal". The same sliding window of 2 weeks is used to calculate the parameters of similarity distribution. Figure 8-4 shows the performance of the TSW-EIA in ROC curves of each alert parameter on the TigerPlace dataset in each plot. Table 8-7 summarizes corresponding AUCs.

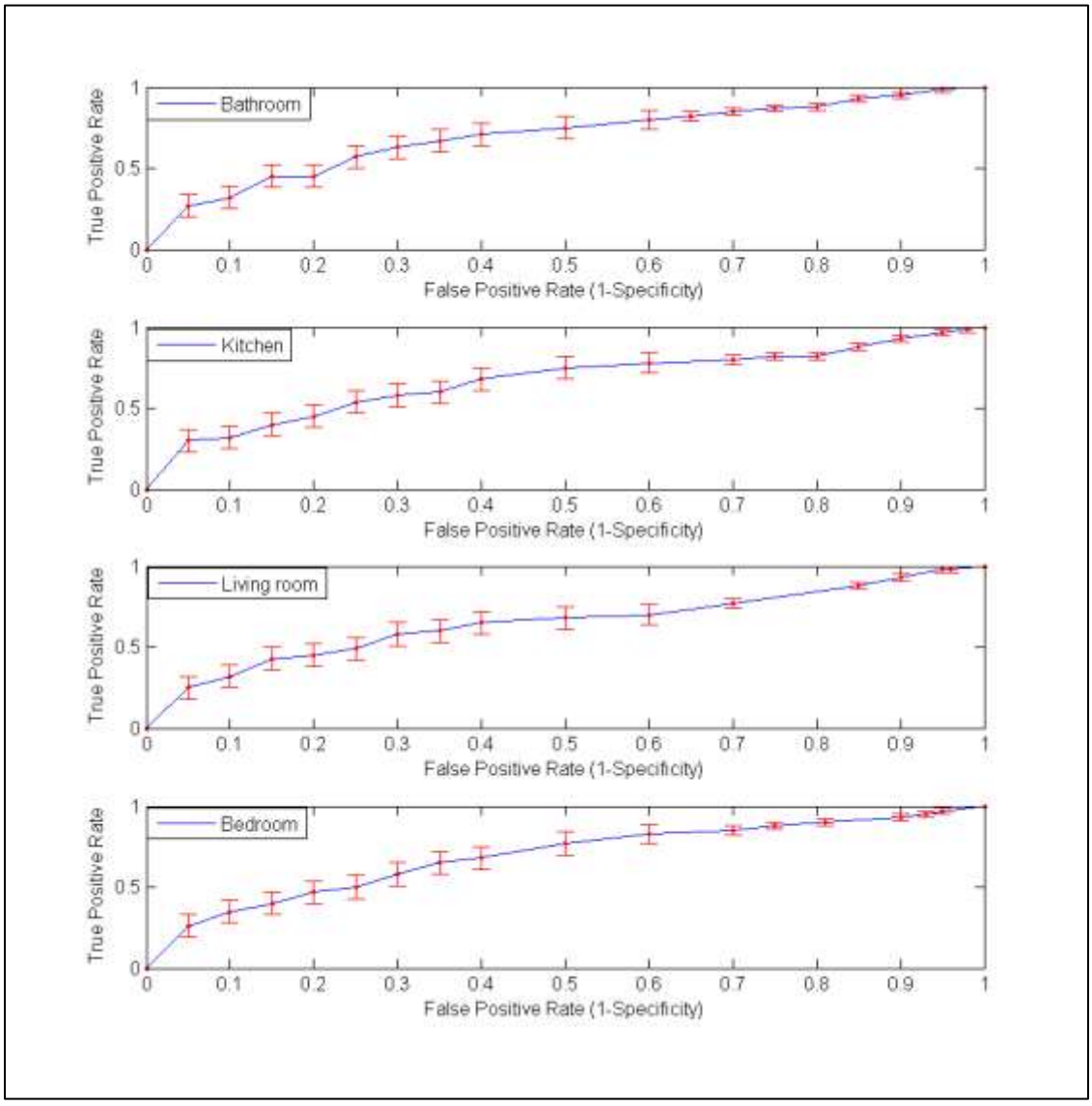


Figure 8-4. EIA using TSW method ROCs on TigerPlace dataset. The plots show the average results on all TigerPlace residents. Error bars are shown in red color indicating variations of results.

Table 8-7. AUC for different activity using TSW-EIA approach

Activity	AUC
Bathroom	0.72
Bedroom	0.70
Kitchen	0.70
Living room	0.69

From the TSW-EIA, the combination of bathroom activity increases at night and during the full day discriminated the best between good and poor user-days, with the same combination except with bedroom performing second best in terms of AUC. This matches more closely with what the clinical researchers recommended. Alert periods for the living room activity resulted in worse AUC than the AUC of other parameters (activities). The reason is usually residents visitors hang out in the living room area which affects the motion sensor log files. Since with the current motion sensors architecture there is no way to identify the source of triggered motion, this seems predictable that living room activities are identified with higher false alarm rates. Moreover, the results of these experiments indicate that those alert parameters (bathroom, bedroom, and kitchen activities) are more clinically relevant, which also agrees with what the clinical researchers have suggested. This set of experiments shows TSW method improved the accuracy of the EIAs to 70% on average over all activities and residents of this study.

8.4. Comparing Early Illness Alert Algorithms

In this section, we summarize the results of TigerPlace EIA method and TSW-EIA. Table 8.8 compares the false alarm rates of EIA methods averaged over four parameters for each resident separately.

Table 8-8. False alarm rate comparison

ResidentID	TigerPlace EIA false alarm rate	TSW-EIA false alarm rate
#1	0.38	0.11
#2	0.27	0.19
#3	0.68	0.23
#4	0.36	0.15
#5	0.47	0.18
#6	0.87	0.26
#7	0.49	0.17
#8	0.52	0.21

The false alarm rates differ across residents in both methods. The functional ability of residents and their health status affects the performance of EIA methods. For example, residents #3 and #6 have highest false alarm rate among other residents. This is because they are the healthiest and younger adults. They do not use walkers, and they are able to perform all their ADLs independently. Therefore, these residents a wide range of different activities with a high variance. They are very active and perform their daily routines with more variations.

However, other residents such as resident #1 and #4 are oldest residents with more health problems who use walkers. They take medications which affect their ADLs such as bathroom visits as well as their sleep patterns. Since they are more dependent on their needs, they are more rigid about their needs. They have certain plans for performing their daily activities such as taking shower. Therefore, they have lower variations in their activity patterns those results in lower false alarm rate.

TSW-EIA out performs the TigerPlace EIA method. The reason is that TSW similarity measure more accurately distinguishes between similar activities, whereas Single dimensional EIA loses the accuracy of detection of abnormal events by aggregating sensor hits. Therefore, the main bottleneck of TigerPlace EIA method is the information loss as a result of data aggregation.

In both methods, Kitchen activities as well as living room activities have the larger number of false alerts, while bathroom and bedroom activities have the lowest number of false alerts. This might be due to the effect of the visitor's motions on the sensor logs. Since the majority of time visitors are hanging out with residents in the living room area, and there is no way to distinguish between different people's activities in the motion sensor data, the false alarms of the living room activities is higher.

Another reason is activities in the living room have the larger variety compare to the activities in the bedroom or in the bathroom. Despite the fact that elderly residents tend to follow almost same routines every day, the healthier and more active they are the more variance they have in their daily routines. This causes more false alarms in the kitchen and living room activities.

Chapter 9 Activity Motif Discovery in Sensor Sequences

As sensor time series data becomes widely available, mining and understanding such data have got more attention from researchers. One of the important tasks in sensor time series mining is finding frequent patterns. More specifically, finding periodicity in time series data is essential in data mining tasks such as prediction and classification. In health monitoring systems using sensor networks, finding the most frequent sensor sequences, is crucial to understand the activity of residents who are under study, and to predict future activities and detect anomalies in their health status. However, periodic behavior in sensor time series could be complicated due to multiple interleaving periods, partial time span, noises, and outliers.

In this chapter, we discuss existing techniques to mine periodic behaviors, known as motifs, from sensor time series data, with a focus on tackling the aforementioned difficulties raised in real applications. In particular, we first review the traditional time-series methods for motif detection. Following, we propose three approaches on motif discovery on sensor time series. We discuss preliminary experimental results using frequent item set mining approach, clustering approach, and statistical approach on TigerPlace dataset. These approaches are used to observe residents activities and detect periodicity on sensor sequences. The periodic sensor sequences are used to annotate sensor sequences and predict any abnormality in resident's health status.

9.1. Introduction

Spatiotemporal data is growing fast with rapid development of sensor networks, positioning technologies, and online social media. Such data can be collected from smart phones, sensor tags attached to animals or installed in the environment, GPS tracking systems on cars and airplanes, RFID tags on merchandise, and location-based services offered by social media. These tracking systems are considered as real-time monitoring platforms and analyzing spatiotemporal data generated from these systems opens many research problems and high-impact applications. For example, using sensor networks technology in animal science may help understanding and modeling animal movement which is important to address environmental challenges such as climate and land use change, bio-diversity loss, invasive species, and infectious diseases [212].

There is an urgent need to develop computational techniques for understanding and mining the ever-increasing amount of sensor data. In this regard, one of the most important research questions is how to find common patterns, more specifically periodic behaviors. In an informal way, a periodic behavior is defined as the repeating activities with respect to a certain location and regular time intervals. For example, employees have weekly periodicity working in their offices. Animals such as eagles have periodic behaviors in their immigration routines.

Mining periodic behaviors has many benefits in different domains. First, discovering periodic behaviors provide an insightful and concise explanation of a long activity history [195]. For example, in health monitoring application the movements of each resident can be summarized using mixture of multiple daily and yearly periodic behaviors. Second, periodic behaviors are also useful for compressing spatiotemporal data [196] [197] [198]. Usually spatiotemporal data becomes a big data as time passes. In an efficient approach, by extract periodic patterns and recording those patterns instead of original data, we will save a lot of storage space without losing much information.

Finally, periodicity is extremely useful in future movement prediction and anomaly detection [199]. For example, if a TigerPlace resident fails to follow regular

periodic behaviors, such as a regular bathroom visit, it could be an indication of an abnormal event due to a change of health status or an accident.

In the next section, we provide a literature review on existing methods and techniques to find periodicity in sensor time series. Later in this chapter, we propose three methods for motif discovery on senor time series including item set mining approach, clustering approach, and statistical approach. We provide experimental results on TigerPlace resident's dataset.

9.2. Related Works

Mining periodic patterns, also known as extracting semantic from raw data, is an open question in data mining society. For example, in health monitoring applications motion sensor data (see Table 2-2) shows the raw movement data of a resident. It is almost impossible to manually extract the periodic behaviors from raw motion sensor data. There are multiple periods and periodic behaviors that may interleave with each other. Difficulties such as vast variations of a given periodic pattern due to different ways of doing the same thing, incomplete observations due to uneven sampling, noise, outliers, and having large portion of missing data due to sensors malfunction complicate the task of mining periodic patterns in sensor time series.

In literature, different periodicity mining techniques have been proposed on various types of data, such as signal processing, gene data, and sensor sequences. In this chapter, we focus on periodicity mining techniques on sensor sequences data. Mining periodic patterns can be divided into two problems: (1) period detection and (2) periodic behavior mining [195]. Period detection is to automatically detect the periods in time series or sequences. Periodic behavior mining problem is to mine periodic patterns with a given period [195]. The periodic behavior mining problem, which is the problem of efficiently locating previously defined patterns in a time series database (also known as query by content), has got much attention from researchers and one can consider that as a solved problem [200] [201] [202] [203] [204]. However, the problem of detecting previously unknown and frequently occurring patterns is still an open question. Authors

in [200] called such patterns motifs, because of their similar analogy to computation biology [205] [206].

Detecting periodic patterns in sensor sequences is different from finding them in real valued time series such as speech signals. In a sensor sequence, we have a series of symbols which represents discrete event. For example, an event can be an X10 motion sensor firing that happened when a person moved in its area of detection. Each event is associated with a time stamp that shows when the firing happened. The problem is to find whether there is an event or a set of events that have periodicity [195]. In health monitoring system such as TigerPlace application, each event is a sensor hit stamped by time. The problem is to detect a sequence of sensor hits that have periodicity. In this scenario, the challenge is that people do not follow the exact same routine to perform the same task. We illustrate this in Figure 9-1 [200]. Figure 9-1 shows an example of different occurrences of a motif in an astronomical database.

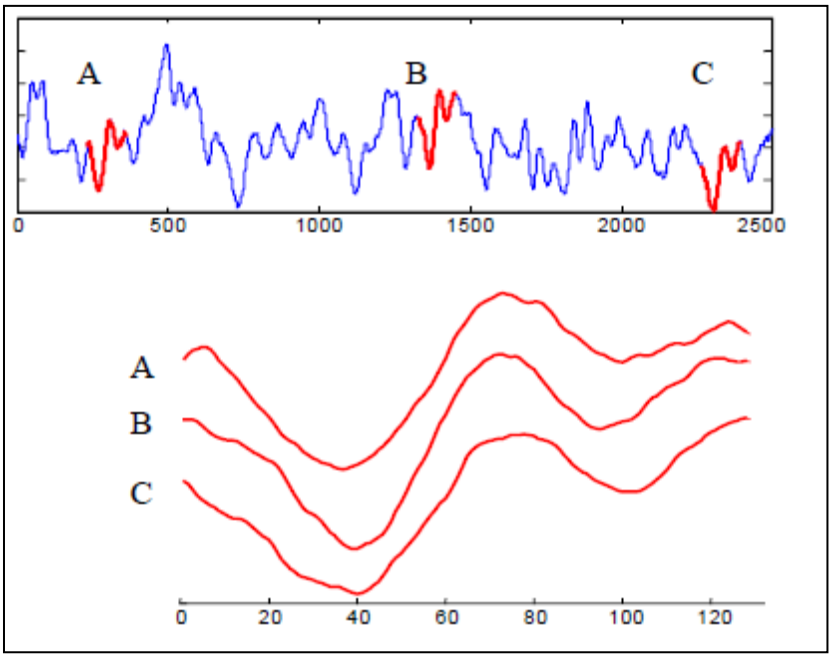


Figure 9-1. Motif discovery in astronomical time series.

To tackle this problem, one approach is to discover periodic patterns using association rule mining [207] [208]. In this approach periodic patterns are referred to as *"frequent patterns"* or *"primitive shapes"*. Even though these approaches are successful on a transactional time series dataset, they either have high quality and are very

expensive or low quality but cheap in sensor time series. We discuss the results of applying this approach on TigerPlace data set later in this chapter.

There are several data mining techniques proposed to discover reoccurring patterns in event streams [209] that are referred to as time series [210]. Event streams are usually sequence of ordered nominal variables that don't have natural ordering. Therefore, the problem of finding periodic patterns in this case castes to finding similar subsets, not similar subsequences [200].

In [211], authors proposed a method to find representative trends in time series. In a similar way, Lin [200] proposed a method that finds frequent patterns, motifs, for time series dataset that can be used summarizing and visualizing massive time series datasets. However, [211] just considered locally representative trends, whereas Lin considered the more globally occurring motifs. Their work had two limitations including poor scalability of the motif discovery algorithm, and weak performance on the presence of noise. In [115], they presented an improved version of their algorithm base on a probabilistic model. The proposed algorithm finds time series motif with a high probability even in the presence of noise or *"don't care"* symbols.

In a weather prediction application [212], authors used labeled data to find frequent motifs. They proposed an efficient approach that finds temporal motifs in multi-dimensional temporal streams of real-world data that are useful for prediction. First, the more important dimensions are selected using SAX [213]. Then the frequent motifs of each dimension are discovered using a sliding window and metrics such as probability of detection (POD) and the false alarm ratio (FAR). POD is the number of times that an event was correctly predicted divided by the total number of observed events.

Floratou, et al. [214] proposed a novel method called Flexible and Accurate Motif Detector (FLAMD). The method is a flexible suffix-tree-based algorithm that finds frequent motifs. They also addressed the problem of extending structure motif extraction which allows mining frequent combinations of motifs under relaxed constraints.

Authors in [215], proposed the first online motif discovery algorithm. The proposed method monitors and maintains motifs in a real time manner.

9.3. Motif Discovery in Healthcare

Characterizing health trajectories of older adults, reflected in patterns of day-to-day activities, may help identify those at the greatest risk for decline and adverse events. Automated in-home monitoring systems may note deviations from daily routines and alert the healthcare providers for timely interventions. Our aim is to develop clinically meaningful methods to discover motifs of daily routines using a currently developed passive sensor network. Frequent sensor sequences (motifs) are used to detect within-person variability of routine activities that may apply as a new predictor in the study of health trajectories of older adults.

The problem of finding motifs in sensor time series may be considered from two points of view. In one approach, we may use prior knowledge as labeled data to guide the motif discovery algorithm. We explain this approach in Figure 9-2 (next page). In this figure, we represent 4 sensor sequences of bathroom visits of a specific resident in TigerPlace. Each bathroom visit is considered as a labeled data point, and the question is how to find the frequent sensor subsequences among them. We propose three approaches to address this problem including item set mining (section 9.3.1), clustering approach (section 9.3.2), and statistical approach (section 9.3.3).

9.3.1. Motif Discovery Using Item Set Mining

Item set mining (a.k.a. frequent item set mining) is a promising solution to discover motifs [216]. This approach identifies frequent item sets discovered in a dataset using different measures of interestingness. Item set mining is well known for market basket analysis which discovers regularities between products in large scale transaction data recorded in super markets. Other applications are web usage mining [217], intrusion detection [218], and bioinformatics.

9.3.1.1. Item Set Mining in Healthcare

In health monitoring application, we use item set mining to discover frequent patterns in sensor time series dataset. To this aim, we use labeled data as represented in Figure 9-2 for explanatory example.

$Visit_1$: Bathroom (10,54,5), Bathroom (10,54,15), Bathroom (10,55,38), Closet (10,55,49)
$Visit_2$: Bathroom (12,24,53), Bathroom (12,25,14), Bathroom (12,25,21), Closet (12,25,51)
$Visit_3$: Bathroom (13,58,49), Bathroom (13,58,56), Bathroom (13,59,12), Bathroom (13,59,29), Bathroom (13,59,39), Bathroom (13,59,46), Bathroom (13,59,55), Bathroom (14,0,3), Closet (14,0,18)
$Visit_4$: Bathroom (17,36,48), Shower (17,36,49), Bathroom (17,36,55), Shower (17,36,57), Bathroom (17,37,3), Closet (7,37,4)

Figure 9-2. Bathroom visits sensor sequences.

In Figure 9-2, we show four bathroom visits of a resident in TigerPlace in a specific day. We manually extracted these sensor subsequences and labeled them as bathroom visits events. Our goal is to develop an algorithm that finds the most frequent patterns among these sequences. One approach is to use item set mining, more specifically Apriori algorithm [219].

9.3.1.2. Apriori Algorithm

Apriori is an algorithm for frequent item set mining proposed by [219]. The method starts with identifying individual items in the database and extending them to larger items in a repeated process as long as the discovered item sets are appeared frequently enough in the dataset. Apriori operates on a transactional database such as a collection of items bough by customers. Each transaction is a set of items, also known as an item set. A threshold *S*, named as minimum support, is used to identify the item sets which are subsets of at least *S* transactions in the database.

In a bottom up approach, Apriori extends frequent item sets one item at a time to generate candidates, and tests the group of candidates against dataset for the minimum support condition. Apriori uses breath-first search approach along with a hash tree structure to count candidate item sets efficiently. The pseudo code for the Apriori algorithm is presented in Figure 9-3, where *T* is a transactional database; *k* is the length of candidate item set, which is the number of items in the candidate item set extracted from item sets of length *k-1*. C_k is the candidate set of length *k*.

Apriori(T,S)

$L_1 \leftarrow \{item\ sets\ of\ size\ 1\}$

$k \leftarrow 2$

$while\ L_{k-1} \neq \emptyset$

$C_k \leftarrow \{a \cup \{b\} | a \in L_{k-1} \wedge b \in \bigcup L_{k-1} \wedge b \notin a\}$

$for\ transations\ t \in T$

$C_t \leftarrow \{c | c \in C_k \wedge c \subseteq t\}$

$for\ candidates\ c \in C_t$

$count[c] \leftarrow count[c] + 1$

$L_k \leftarrow \{c | c \in C_k \wedge count[c] \geq S\}$

$k \leftarrow k + 1$

Return $\bigcup_k L_k$

Figure 9-3. Apriori pseudo code.

We propose a novel algorithm to find frequent sensor sequences in TigerPlace dataset based on Apriori method. In the next section, we describe this method.

9.3.1.3. Detecting Motifs Using Apriori and TSW

In this section, we describe our approach to use Apriori and TSW to find motifs or frequent sensor subsequences on TigerPlace dataset. We start with definitions to formulate the problem.

Definition 9.1. *Item:* an item is a window (or subsequence) of size p of a sensor sequence of size n. We represent item I_j by $I_j=\{(s_{jx}\ t_{jx})(s_{j(x+1)}\ t_{j(x+1)})\dots(s_{j(x+p-1)}\ t_{j(x+p-1)})\}$ where $1 \leq x \leq n-p+1$.

Definition 9.2. *Non overlapping Items*: two items $I_i=\{(s_{i1}\ t_{i1})(s_{i2}\ t_{i2})\dots(s_{im}\ t_{im})\}$ of length m and $I_j=\{(s_{j1}\ t_{j1})(s_{j2}\ t_{j2})\dots(s_{jm'}\ t_{jm'})\}$ of length m' are called non overlapping items, if there is no overlap in the time they occurred, that is either $t_{im} < t_{j1}$ or $t_{jm'} < t_{i1}$.

Definition 9.3. *Transaction*: A transaction *T* of size *m* is a set of *m* non overlapping items that occurred sequentially. We show transaction *T* by, *T={I₁ I₂ … Iₘ}* where $I_1 = \{(s_{11}t_{11})(s_{12}t_{12}) \dots (s_{1n_1}t_{1n_1})\}$, $I_2 = \{(s_{21}t_{21})(s_{22}t_{22}) \dots (s_{2n_2}t_{2n_2})\}$, …,

$I_m = \{(s_{m1}t_{m1})(s_{m2}t_{m2}) \dots (s_{mn_m}t_{mn_m})\}$, and $t_{1n_1} < t_{21}$, …, $t_{m-1n_{m-1}} < t_{m1}$.

Definition 9.4. *Candidate set*: a candidate set C_k of size *k* is the set of *k* items.

We propose a new version of Apriori algorithm using TSW in Figure 9-4, named as Apriori_TSW.

Apriori_TSW(T,S, ε, δ)

$L_1 \leftarrow \{item\ sets\ of\ size\ 1\}$

$k \leftarrow 2$

$while\ L_{k-1} \neq \emptyset$

$C_k \leftarrow \{I_i \cup \{I_j\} | I_i \in L_{k-1} \wedge I_j \in \cup L_{k-1} \wedge TSW(I_j, I_i) < \varepsilon\}$

$for\ transations\ t \in T$

$C_t \leftarrow \{c | c \in C_k \wedge \forall b \subseteq c, a \subseteq t, TSW(b,a) > \delta\}$

$for\ candidates\ c\ \in C_t$

$count[c] \leftarrow count[c] + 1$

$L_k \leftarrow \{c | c \in C_k \wedge count[c] \geq S\}$

$k \leftarrow k + 1$

Return $\cup_k L_k$

Figure 9-4. Apriori_TSW algorithm pseudo code.

Apriori_TSW finds motifs on sensor sequence transactions based on three parameters S,ε, δ. Parameter S controls the minimum support threshold in the original Apriori algorithm while parameter ε controls the threshold of similarity of two items (sensor subsequences). If TSW similarity of two items is less than ε, they are considered as different items and may be used to generate new candidate set. Parameter δ is used as a threshold for counting the number of occurrences of an item of a candidate set in the

transaction dataset. In the next section, we present experimental results of Apriori-TSW method on TigerPlace dataset.

9.3.1.4. Experimental Results Using Apriori-TSW Method

To evaluate the performance of Apriori-TSW method, we used the labeled sensor sequences extracted from a resident's history of activities in TigerPlace. We manually extracted all bathroom visits of one week (the first week of January 2013) of a particular resident. Three sensors are involved in bathroom visits activity including *Bathroom*, *Shower*, and *Closet*. Table 9-1 presents the statistical information of the dataset used for this method and Figures 9-5, 9-6, and 9-7 visualize this information in graphs.

Table 9-1. The statistics of bathroom visits of a TigerPlace resident

Number of bathroom visits	72
Average bathroom visits duration in seconds	132
Average number of sensor hits for bathroom visits	4

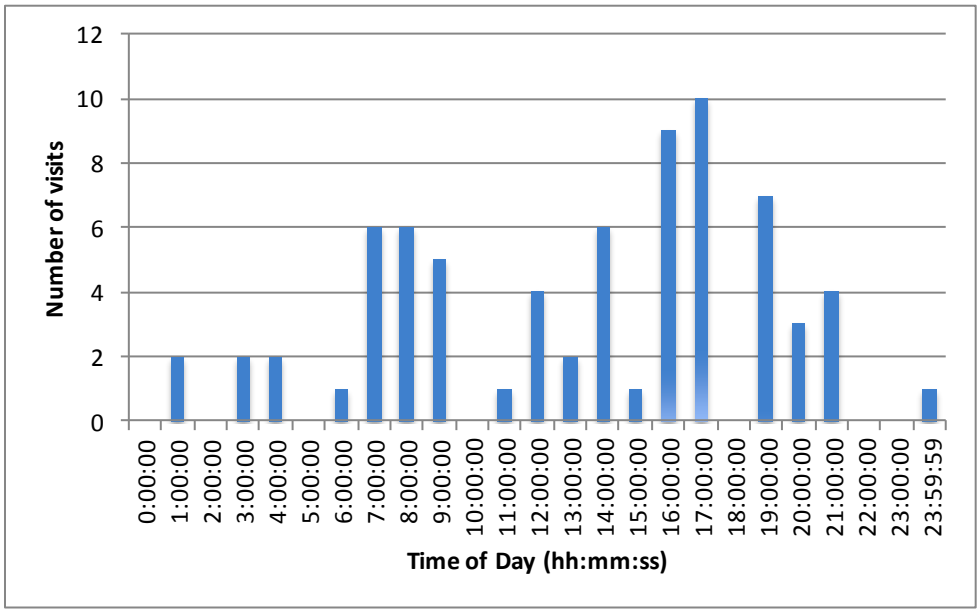


Figure 9-5. Number of bathroom visits on different times of days.

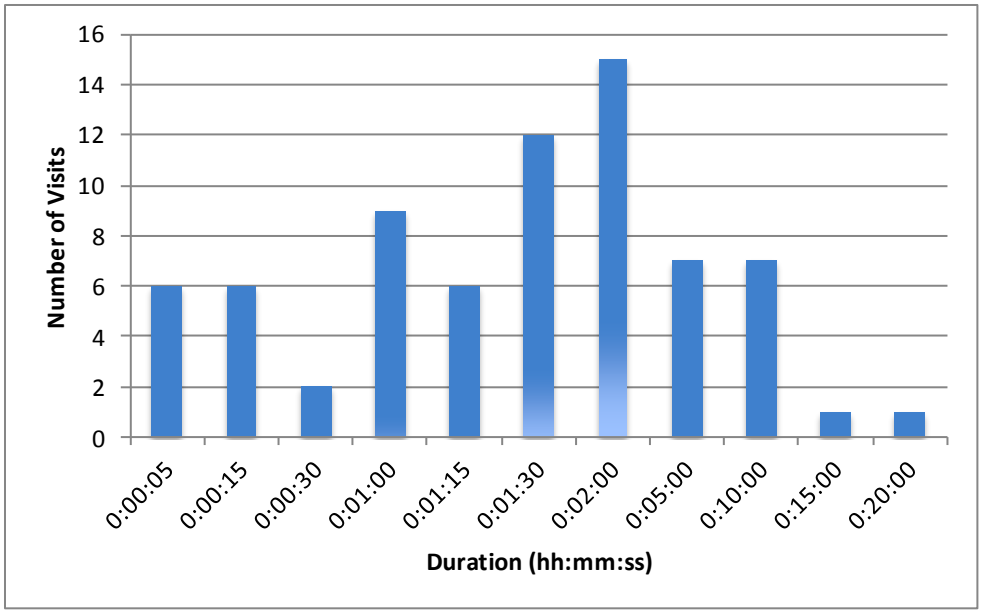


Figure 9-6. Number of bathroom visits with different duration of the visit.

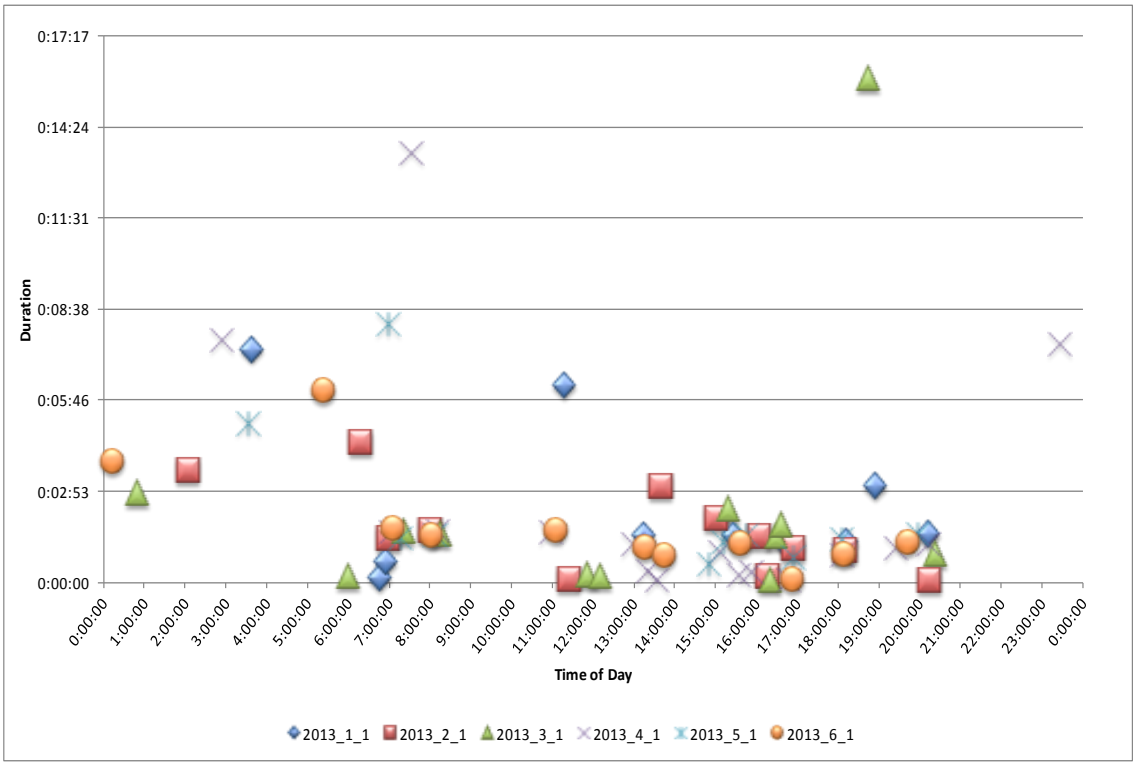


Figure 9-7. Duration and occurrence of bathroom visits on 6 different days for a TP resident.

We examine the performance of Apriori-TSW on data shown in Table 9-1. We first explore the effect of parameter initialization on the performance of Apriori-TSW. Figure 9-8 shows the number of extracted frequent item sets (FIs) based on the size of an item that is parameter *p* (see definition 9.1). In this Figure, the parameter *p* represents the size of an item that is the number of sensor hits in the item. Increasing the size of an item decreases the number of frequent patterns. Based on manual verifications by our experts, we set *p=5* for the rest of our experiments.

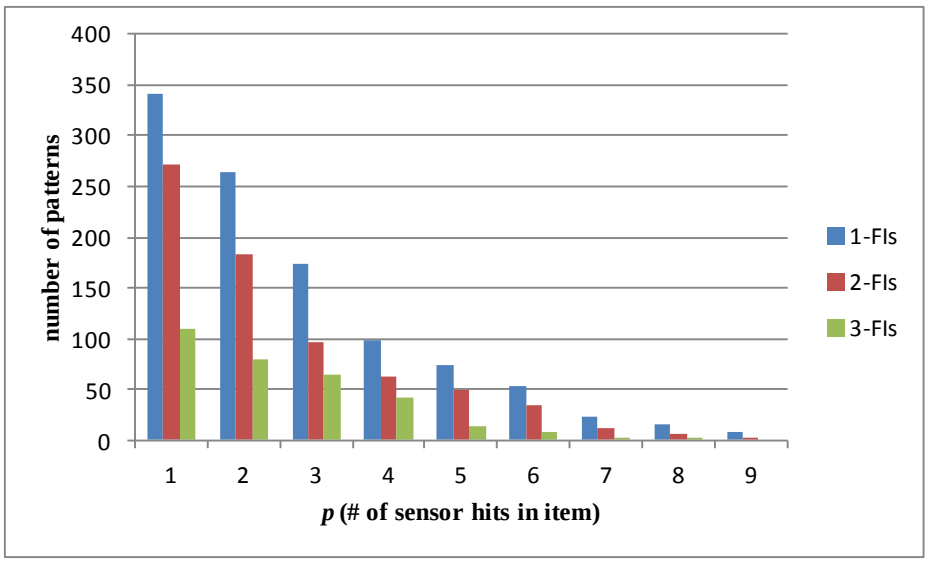


Figure 9-8. The effect of parameter *p* on Apriori-TSW algorithm.

Figure 9-9 shows the results of initializing parameter δ. This parameter is used in Apriori-TSW to control the level of similarity between two items. If the similarity of two items is greater than the parameter δ, it will be considered as occurrences of the same item in the transaction dataset and increments its frequency by one. As the value of parameter δ increases the number of FIs decreases.

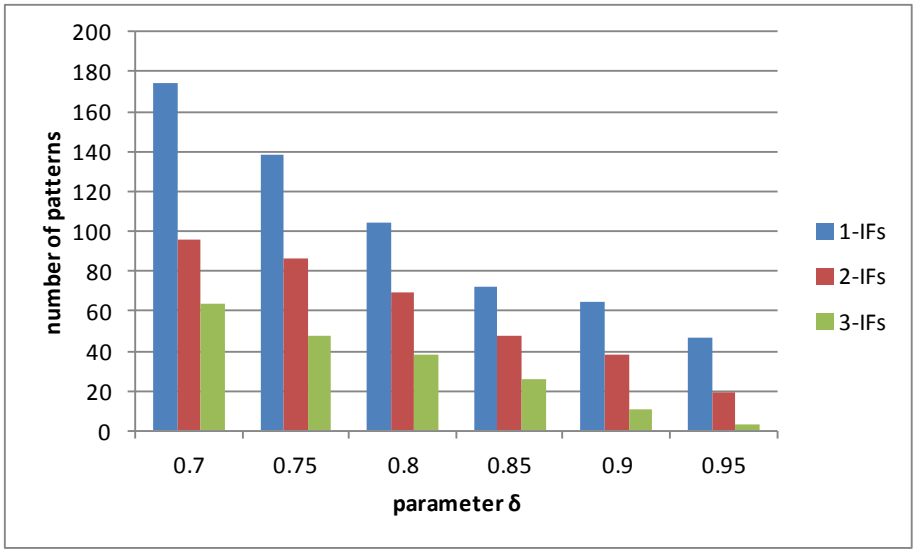


Figure 9-9. Effect of parameter δ on the number of frequent patterns.

Parameter ε in Apriori-TSW is used to control the level of dissimilarity between two different items. Figure 9-10 presents the number of different patterns extracted based on different initial values of this parameter.

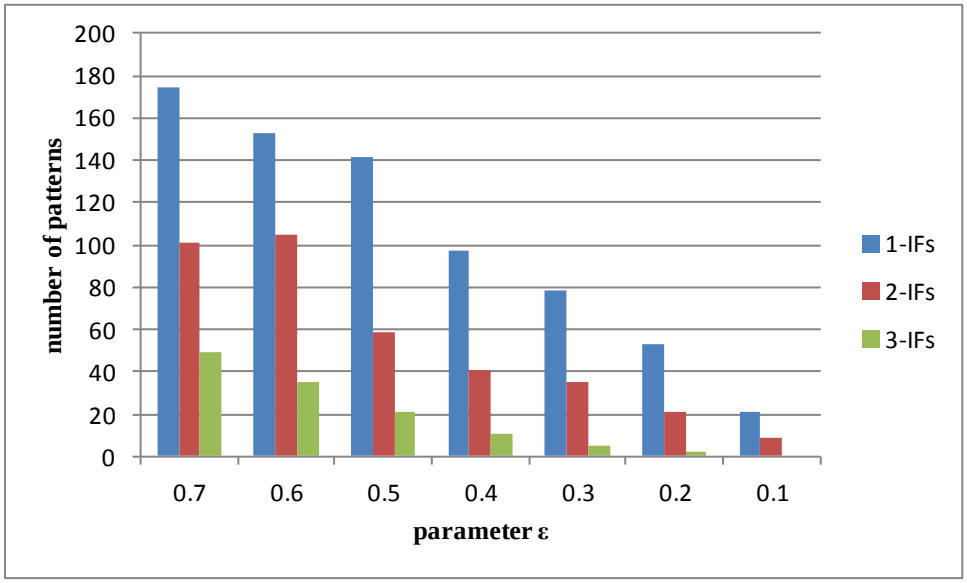


Figure 9-10. Initializing parameter ε.

9.3.1.5. Limitations of Apriori-TSW Method

We proposed Apriori-TSW as a new solution to find motifs in sensor sequences. However, this method has the following limitations:

1) The method is verified on the labeled data that is not available in most applications.
2) Defining an item is a challenging task in this method. If we consider each sensor event (a sensor hit) as an event, then the time complexity of this method is high where shorter item sets (item sets of size one, two or even three) do not represent a meaningful activity. On the other hand, representing an item as a sensor sequence (sensor hits within one minute for example) has more challenges such as initializing the size of window, chunking the dataset, etc.
3) On large data sets, this method can be very slow and does not support online monitoring for upcoming activities.

To address some of difficulties of Apriori-TSW method we proposed a clustering approach on motif discovery on sensor time series.

9.3.2. Motif Discovery Using Clustering Approach

Functional ability can be reflected in the performance of routine tasks that include activities of daily living (such as bathing, dressing, hygiene, and bowel movement), more complex instrumental activities of daily living (s. a. housework, finance management, and shopping), and general life activities (s. a. hobbies, leisure past time and socialization). Routines develop within temporal patterns of day and night, weekday and weekend, and seasonal change [3]. Extracting and analyzing attributes of daily routines may be useful in understanding trajectories of functional decline of older adults and can suggest timely interventions. In sequences of nominal variables that do not have natural ordering, such as our sensor data, the goal is to find analogous subsets, or “primitive shapes”, rather than similar subsequences. In this section, we propose a novel computational approach for mining periodic patterns of bathroom visits, as part of the

daily routine, based on the similarity of sensor firing sequences (SFS) and frequent SFS patterns.

9.3.2.1. Identifying Frequent Patterns on Bathroom Visits

We focus on bathroom visits, as an important daily routine, and extract three features: duration, number of sensor hits, and the average time between hits. We use distributions of these parameters to define the confidence level for validating the automated bathroom visit extraction process. Then, a hierarchical clustering approach is applied to cluster visits and extract frequent bathroom routine patterns.

9.3.2.2. Automated Routine Extraction Process

We apply a rule-based approach to automatically extract bathroom visits from sensor log files. The rules are based on the apartment floor maps and geospatial locations of the motion sensors in each apartment. Then, we apply a hierarchical approach to validate the extracted visits.

Given n days of data, assume the automated process extracts $\{m_i\}_{i=1,n}$ bathroom visits for day i. Let's denote the k^{th} visit, $1 < k < m_i$, of day i by a sensor sequence $T_k=\{(C_{k1}\ t_{k1}), (C_{k2}\ t_{k2}),\ldots,(C_{kp}\ t_{kp})\}$ where p is the length of the visit, $p \in N$, C_{ij} belongs to alphabet Σ, and t_{ij} are the time of the day of the firing C_{ij}. For each visit we define three parameters as duration of the visit in seconds (D), length of the visits that is the number of sensor hits in the visit (S), and the average time difference between consecutive hits in the visit (H_{AVG}) as formulated in equations 9-1 to 9-3.

$$D = | t_{kp} - t_{k1} | \quad (9.1)$$

$$S = p \quad (9.2)$$

$$H_{AVG}=\left(\left(\sum_{i=1}^{p-1} |t_{k(i+1)} - t_{ki}|\right)/p\right) \quad (9.3)$$

Assume we found out that parameters S, D, and H_{AVG} follow a *"LogLogistic"* distribution with parameters σ, μ, i.e $\{F_a(\sigma_a, \mu_a)\}_{a=S,D,H_{AVG}}$. The confidence that x with parameters $(x_S, x_D, x_{H_{AVG}})$ is a valid bathroom visit, $C(x)$ is 0 if $\{x_a < \min\ (F_a(\sigma_a, \mu_a))\}_{a=S,D,H_{AVG}}$, and equal to $\{Average(F_a(\sigma_a, \mu_a))\}_{a=S,D,H_{AVG}}$ where F_a is the likelihood that x_a comes from $F_a(\sigma_a, \mu_a)$ and is calculated using:

$$F_a(x_a, \sigma_a, \mu_a) = \frac{e^z}{\sigma_a x(1+e^z)^2}, \quad (9.4)$$

where $z = \frac{\log(x)-\mu_a}{\sigma_a}$.

9.3.2.3. Hierarchical Clustering on Routines

After retrieving valid bathroom visits, we use a hierarchical clustering approach to find repeated patterns. In the clustering process we use TSW to measure the similarity of two sensor sequences. We use the "time of the day" metric, since we would like to find similar routine behaviors across different days that happen at roughly the same time.

Applying agglomerative hierarchical clustering, we seek to build a hierarchy of bathroom visit clusters. In this bottom-up approach, each observation is considered as a cluster, and pairs of clusters are merged as one move up the hierarchy. Figure 9-11 illustrates the hierarchical clustering with TSW.

Algorithm: Hierarchical Clustering using TSW

Input: *N* sensor sequences, and an *N*N* similarity matrix;

Output: a dendrogram representing the hierarchical structure;

Steps:

1. Consider each sensor sequence as its own cluster.
2. Find the most similar pair of sensor sequences using TSW and merge them into a single cluster.
3. Compute distances between clusters as follows $D(C_i,C_j)=max\{1-TSW(S_{ik},S_{jk'})\}_{k=I,n,k'=I,m}$ where cluster *i* has *n* and cluster *j* has *m* sensor sequences.
4. Repeat steps 2 and 3 until all sensor sequences are clustered into a single cluster of size *N*.

Figure 9-11. Hierarchical clusterig using TSW algorithm.

9.3.2.4. Pilot Dataset

Our sample consists of three residents with different bathroom habits. Their characteristics are described in Table 9-2. Resident #1 does not have any reported urinary problems, but takes diuretic medications for high blood pressure. It may alter his bathroom habits, causing more frequent and varied visits. Resident #2 is diagnosed with urinary retention from an enlarged prostate and has difficulty emptying the bladder. Resident #3 is occasionally incontinent. All three take laxatives to improve bowel motility.

Table 9-2. TigerPlace Resident Characteristics

Resident ID	Age	Gender	Urinary Problems	Ambulates with Walker	Diuretic Medication
#1	88	Male	None	No	Yes
#2	99	Male	Retention	Yes	No
#3	90	Female	Incontinence	Yes	No

Table 9-3 shows an example of a bathroom visit as recorded by our in-home monitoring system. As the resident moves through the apartment, the motion sensor firing data is added to the log file database. In this example, on October 5, 2013, around 12:30AM Resident #3 was in the bathroom for about 3 minutes. The visit begins with the first bathroom sensor hit and ends when the person enters another room (walk-in closet). The sequence has the length of 5 sensor hits.

Table 9-3. Example of sensor firings for a bathroom visit

Resident ID	Sensor	Year	Month	Day	Hour	Min	Sec
#3	Bathroom	2013	10	5	12	34	38
#3	Bathroom	2013	10	5	12	36	52
#3	Bathroom	2013	10	5	12	37	04
#3	Bathroom	2013	10	5	12	37	11
#3	Closet	2013	10	5	12	37	26

Table 9-4 shows the total number of known bathroom visits from the apartments of the three residents for the period of 10 consecutive days used in this study.

Table 9-4. TigerPlace pilot dataset

Resident ID	Number of bathroom visits	Number of shower visits
#1	74	3
#2	82	4
#3	69	1

Because routine behaviors are person-specific, mathematical models cannot be transferrable from one resident to another. Hence, we view our data as three separate experiments with more than 200 samples of bathroom visits, rather than one experiment with only three samples.

We run a set of experiments to investigate the performance of the proposed routine detection based on frequent sensor sequences. Figure 9-12 shows three scatter plots of the duration and length of bathroom visits in terms of number of sensor hits in each visit (shown by their IDs in x-axis) presented in Table 9-4.

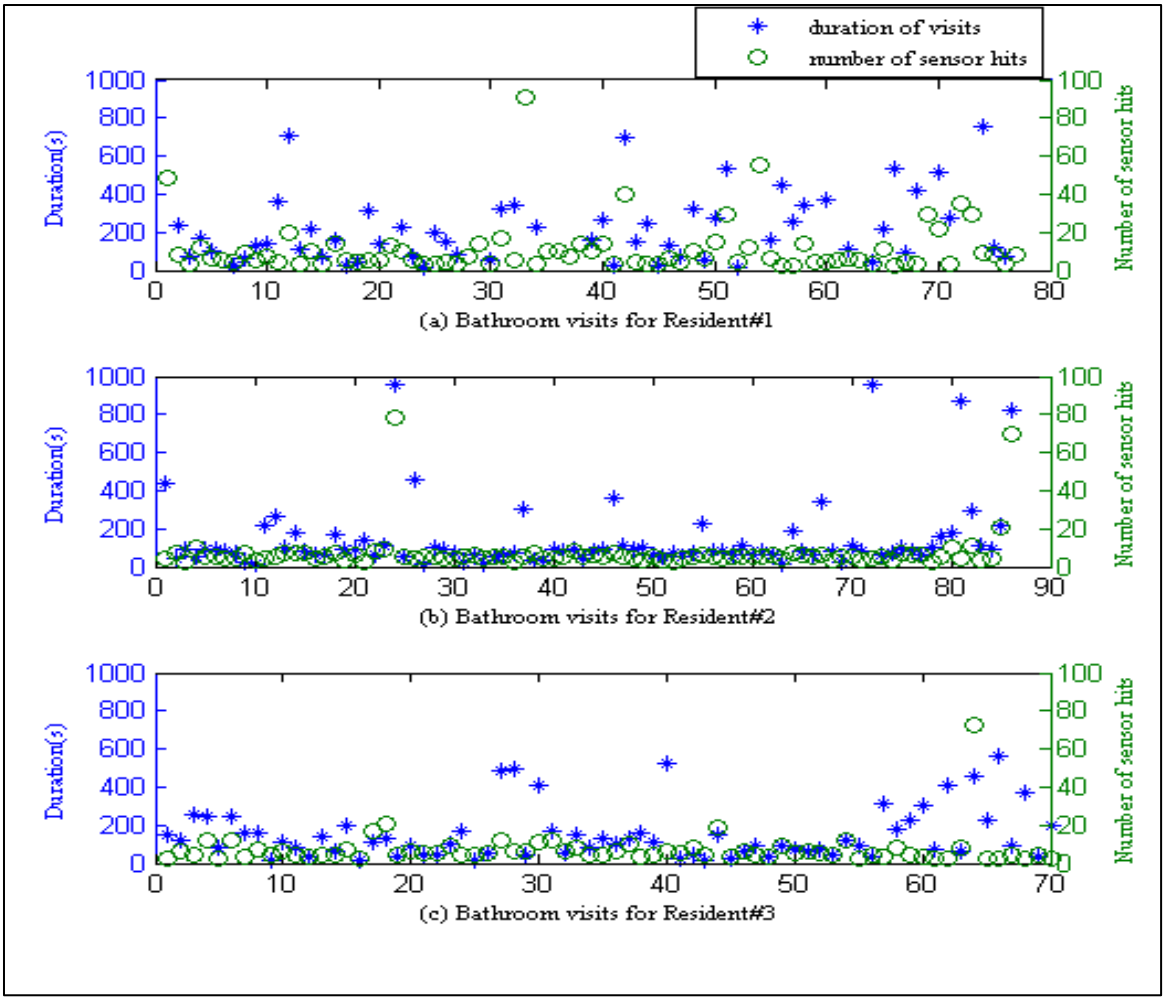


Figure 9-12. Bathroom visits scatter plots.

Bathroom visits range in duration from few seconds to about 16 minutes. The average duration for bathroom visit across people was 2.5 minutes (150 s) and the average number of sensor hits is 7.8. The average duration of for Resident #1 was 3.5 min (214.9 s), 1.87 min (112.5 s) for Resident #2, and 2.8 min (149 s) for Resident #3. While the majority of bathroom visits are shorter, possibly indicative of regular bowel movements, there are a few outliers that may represent more complex hygiene and showering. It is evident that each resident has their own pattern and distribution of visits.

9.3.2.5. Automated Bathroom Routines Extraction Results

To verify the performance of the extraction process on the dataset presented in Table III, we applied the 10 cross-fold validation approach. We used a library of manually extracted sensor sequences of bathroom visits to calculate the parameters of the ROC curve, as shown in Figure 9-13. Table 9-5 presents AUC separately for each of the residents.

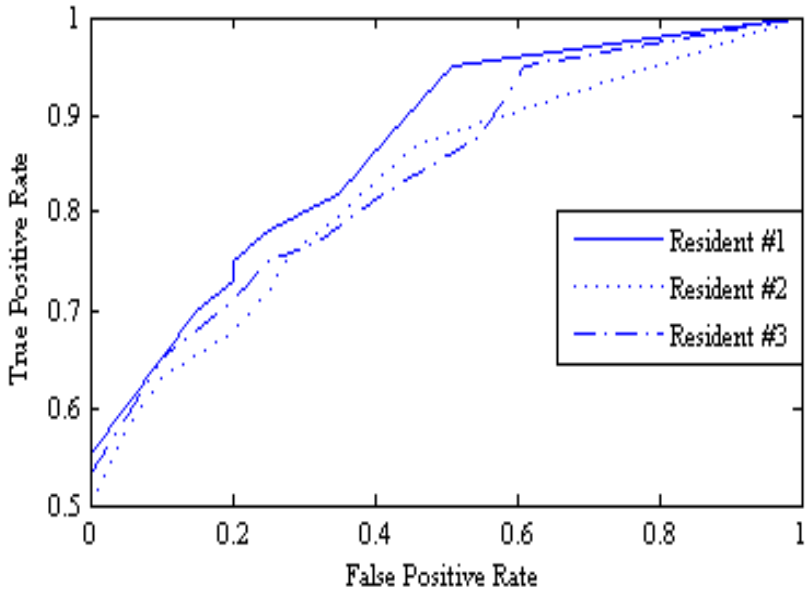


Figure 9-13. ROC curve of automated bathroom visit extraction.

Table 9-5. Performance of the automated bathroom extraction process based on AUC

AUC #1	AUC #2	AUC #3
0.86	0.83	0.84

The average recognition performance is 0.84. Residents #2 and #3 use a walker that affects their speed reflected in the sensor hits. Therefore, the extraction process has a lower specificity, compared to Resident #1. Modeling walk patterns with walker for elderly resident is a complicated task that increases the false alarm rate of our model.

9.3.2.6. Hierarchical Clustering Results on Bathroom Routines

We initialized the parameters of TSW as g=0 and c=0.028, setting the relevant time interval at about 35 seconds. Figure 9-14 visualizes the results of the hierarchical clustering using (1-TSW) as a distance measure. The color map visually represents the strength of the relationships between bathroom visits for each resident separately. We can see that Resident #1 has a large number of small clusters, while Resident #2 has at least two large and distinct clusters (red and blue rectangles). Meanwhile Resident #3 has a mix of large and small clusters.

The number of routine clusters may be related to the functional status of the older adult. Resident #1 is the youngest and most active, which allows him to have a less rigid routine, with bathroom visits that are more varied in time. Also he is the only one taking a diuretic medication that promotes urine production, and in turn may affect the quality of bathroom visits. Both Residents #2 and #3 have urinary problems and may be more mindful of their needs. They use walkers that may force them to have more regimented and planned bathroom trips.

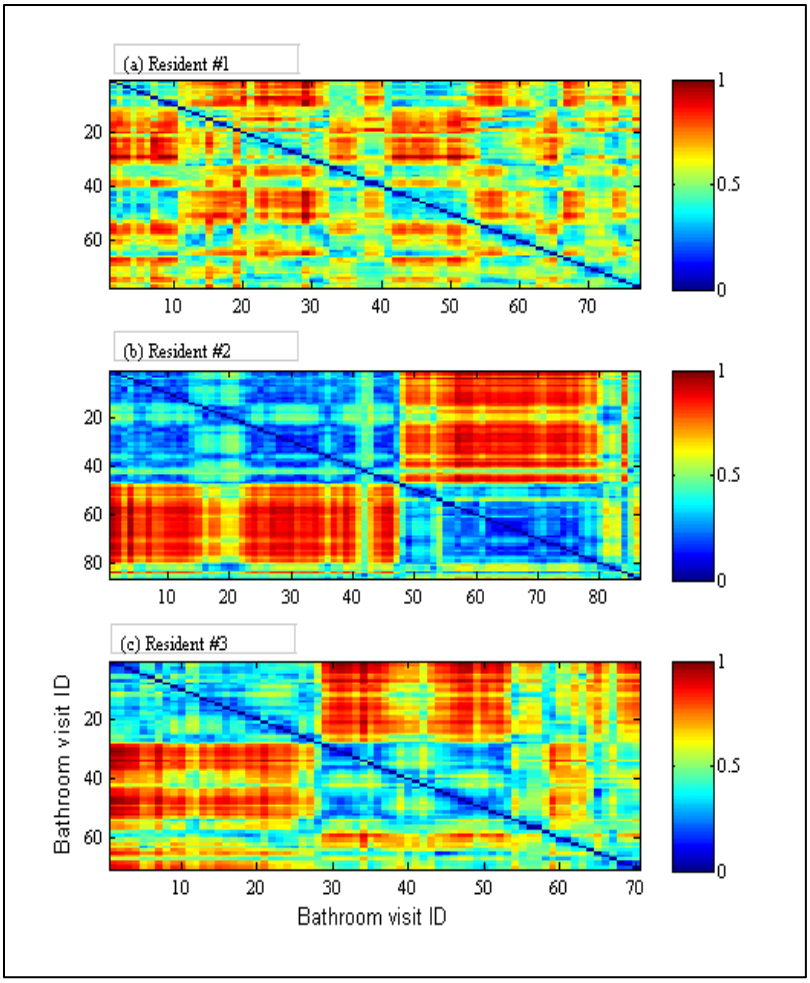


Figure 9-14. Hierarchical clustering results.

In Figure 9-15, the dendrogram shows clusters of bathroom visits based on parameters *D*, *S* and H_{avg} from equation 9-3 (the height of each U represents the distance between the two objects being connected). Based on these characteristics we can identify types of bathroom visits and predict future changes. For example, for Resident #3 the first cluster (dashed red box) includes approximately 1.5 minute visits that happen in the afternoon when the resident is more active. Now that we validated the extraction and identification algorithm, we can explore this relationship between routines and functional abilities with a larger number of residents in future studies.

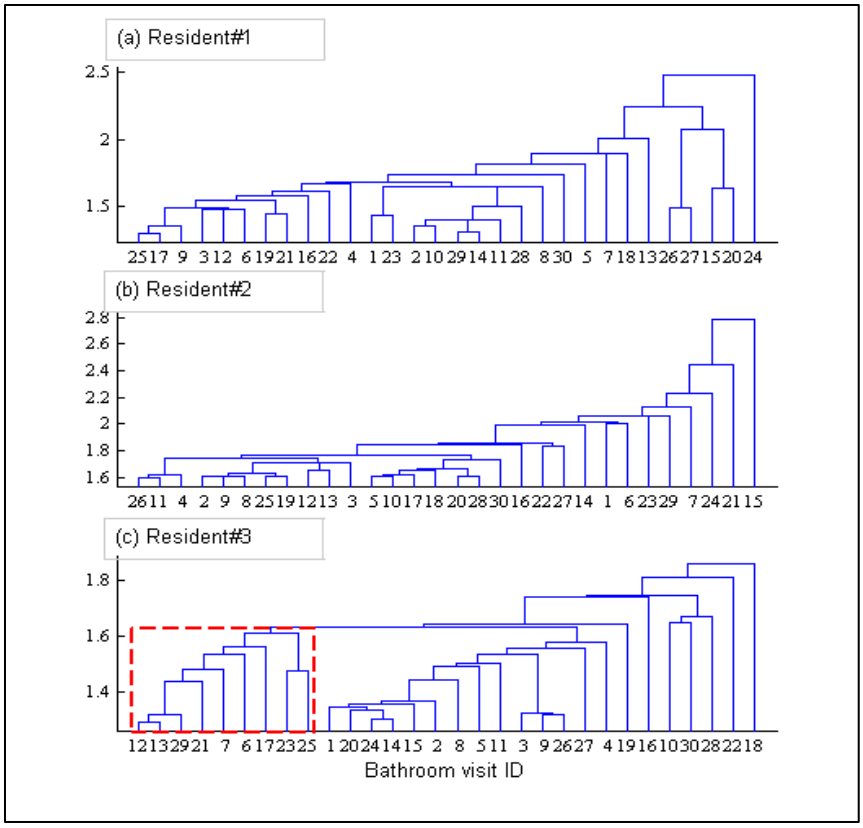


Figure 9-15. Hierarchical dendrograms.

9.3.2.7. Discussion on Motif Detection Using Clustering Approach

Our computational algorithm successfully extracted routine bathroom visits of three TigerPlace residents. Using a hierarchical clustering approach and TSW similarity measure, we detected frequent sensor sequences with an average recognition rate of 0.84, supporting the robustness of the proposed method. Identified clusters of routine behaviors may be related to the functional status of the older resident. Our approach tackles the problem with the big data collected by in-home monitoring systems by conceptualizing target periodic patterns as daily routines. This algorithm can be used to detect other routines, such as mealtimes and social activities. The proposed method can be incorporated into the existing alert system for the healthcare providers to recognize early functional changes reflected in deviations from daily routines.

The limitations of this approach are:

- Once a decision is made to combine two clusters, it cannot be undone

- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
    - Sensitivity to noise and outliers
    - Difficulty handling different sized clusters and convex shapes
    - Breaking large clusters

In the following section, we propose a statistical motif discovery approach to address some of the before mentioned challenges.

9.3.3. Motif Discovery Using Statistical Approach

Typical early illness recognition approaches are either concentrated on the detection of a given set of activities such as a fall or walks, or on the detection of anomalies such as too many bathroom visits. In the rest of this chapter, we propose a new illness recognition framework, MFA, based on detecting a missing frequent activity from the daily routine. MFA is implemented using a frequent temporal pattern detection algorithm and demonstrated on our TigerPlace pilot dataset. Under this framework, at the end of the day, we examine if all frequent patterns found in previous two weeks are still present. If some patterns are found missing several days in a rows, we assume that it is due to an early onset of a decline and we send an alarm to caregiver. To detect the loss of an existent ability, we need to employ a frequent pattern detection method that is independent of the types and number of activities performed by the resident.

9.3.3.1. Temporal Patterns Mining

Many techniques for detecting frequent patterns in time series have been reported in literature such as a combination of suffix tree and Markov model [6], a random projection algorithm [7], a rule based approach using likelihood criterion [8], FP-tree based approach using extended prefix-tree structure [9], fuzzy association rule mining approach [10]. Other frequent pattern detection work is related to activity recognition. For example, in [11] Bayesian networks are used to detect and predict the action time for 11 human activities using motion sensor data, while in [16] Latent Dirichlet Allocation is used for activity recognition using Kinect. However, most of these methods assume a certain number (fixed) of known activities. Moreover, in many cases, the training algorithm process requires data labeling, which is not feasible in a real environment. An implicit activity recognition approach based on sensor sequence similarity that combined sensor and EMR to provide early illness recognition was explored in [14]. One of the most used frequent pattern detection algorithm in behavior sciences is based on T-Patterns [12].

The frequent T-Pattern, FTP, algorithm was first introduced by Magnusson in [12] to extract frequent human behaviors and improved in [13] for artificial sensor data. T-patterns are represented by symbols and time stamps. FTP algorithm finds possible relationships between pairs of symbols by building trees of temporal dependencies. The existent methods [12, 13] consider all possible combination of symbols to form patterns. This approach is potentially slow in sensor networks application for real time monitoring, especially for large number of sensors and long time line. We adapt the FTP algorithm introduced in [12], for detection of frequent patterns in monitoring sensor networks and explore its suitability for MFA on a TigerPlace pilot data set. Following, we briefly describe our proposed method.

In T-Pattern approach [12], Magnusson proposed the notion of a critical interval (CI) to find strongly correlated events (sensor firings). In this approach, all possible combinations of events are considered. The number of CI searches for a single pass of the T-pattern algorithm is $O(n^2h^2)$, where n is the number of event types (or sensors) and h is the event horizon (sequence length). Our goal is to modify this method such that it is

linear in h, O(n^2h), and be suitable for real time processing (we would like to find the pattern in real time as opposed to at the end of the day).

9.3.3.2. Data Representation

We define a time series S as a set of n couples (s_i t_i), i.e. S={(s_1 t_1), …, (s_n t_n)}. Each couple (s_i t_i) , $1 \leq i \leq n$, has two components: a sensor signal s_i that belongs to a symbols set Σ and a time stamp t_i which represents the time t_i when s_i was recorded. The alphabet Σ is a set of identifiers we use to represent multi-dimensional sensor time series. In our experiments, Σ comprises of three symbols as B (for bathroom sensor), C (for closet sensor), and A (for Shower sensor).

9.3.3.3. T-Pattern Representation

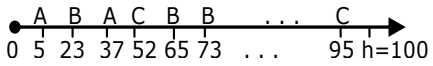
The T-Pattern is defined, recursively, as an ordered m triples, i.e. T={(s_1 $[d_1,d_2]_1$ s_2), …, (s_i $[d_1,d_2]_i$ s_{i+1}), …, (s_{m-1} $[d_1,d_2]_m$ s_m)}. Each triples (s_i $[d_1,d_2]_i$ s_{i+1}) , $1 \leq i \leq m$, has three components: two sensor signals s_i and s_{i+1} that belongs to the symbols set Σ, and an interval $[d_1,d_2]_i$ which reflects different average values of occurences of the same pattern within the observation timeline. In this presentation the term (s_i $[d_1,d_2]_i$ s_{i+1}) reads: when sensor s_i triggers between d_1 and d_2 time units later sensor s_{i+1} triggers.

9.3.3.4. Critical Intervals

For every frequent T-Patterns there is a CI that represents the relationship between the distributions of the elements of the T-Pattern, i.e. sensors [12]. For example, for a T-Pattern (s_i, s_j), CI $[d_1,d_2]$ indicates that if s_i occures at time t then there is an interval $[t+d_1,t+d_2]$ that tends to contain at least one occurrence of s_j.

9.3.3.5. Finding Frequent T-Pattern using CI

Inspired from KMP string search algorithm [15], we propose an efficient algorithm to find T-Patterns without the need exaoustive search of all sensor combinations. For the description of the original algorithm the reader is refered to [1]. Our proposed method has three main steps as described in the following.



Sensor	Time stamp
A	5, 37, …
B	23, 65, 73, …
C	52, … , 95

Figure 9-16. An example of occurrence tables.

- *First Step: Building Occurrence table*

Considering the training observation interval [0,*h*], the occurrence table is a data structure that indexes the firing time of each sensor in the observation interval. Assume we have *n* sensors, then the occurrence table have *n* rows each for one sensor. It takes just one pass through the whole dataset to build this matrix. So the time complexity of this step is *O(h)*. Figure 9-16 shows this step in an example.

- *Second Step: finding couple T-Patterns*

Having the occurrence table, we search for 2-TPatterns (patterns of size 2, such as pattern "CB"). To find a 2-TPattern (s_i s_j), the algorithm starts at row *i* in the occurrence table. Then for each occurrence of s_i at time t_p it seeks for the first occurrence of s_j at time t_q such that $t_q>t_p$, where $1<p<N_i$ and N_i is the number of occurrences of s_i in the observation time, and $1<q<N_j$ and N_j is the number of occurrences of s_j in the observation time interval. Having two occurrences of the 2-TPattern (s_i s_j) at [t_p,t_q]$_1$ and [t_p,t_q]$_2$ the related CI is formed using the size of each occurrence that is [Δd_1,Δd_2] where Δd_1 is the size of [t_p,t_q]$_1$ in time units. After finding all CIs, we sort them increasingly based on the size of CI. Then we perform a *p*-test to verify the significance of the CI. The *p*-value of CI [d_1,d_2] is calculated using equation 9-5.

$$p = \frac{(1-\sum_{k=0}^{N_{i,j}-1} binomial\left(N_{i,j},k,P(\sim s_j)\right))}{N_{i,j}} \qquad (9.5)$$

Where $N_{i,j}$ is the number of occurrences of pattern (s_i s_j), and $P(\sim s_j)$ is the probability of absence of sensor s_j, $P(\sim s_j) = 1-(frequency(s_j)/h)$. Intuitively, if the CI of size Δd_1 is not significant, then the CI of size Δd_2 where $\Delta d_2>\Delta d_1$ is not significant either. The time complexity of this step in the worst case is $O(nh)$ and in the best case is $O(n^2)$ where *n* is the number of sensors.

- *Third Step: Finding k-TPatterns*

In this step in an iterative process we find T-Patterns of size *k*, called as *k*-TPatterns. For a given (*k-1)*-TPattern ($2>k$, $k \in \mathbb{N}$) with significant CI [d_1,d_2], the next *k*-TPattern is built by looking up in the occurrence table for the first sensor event at time t_q where $t_q>t_p$ and t_p is when the last event when the (*k-1)*-TPattern happened. This step has the same time complexity as the second step. Figure 9-17 shows the pseudo code of Frequent T-Patterns algorithm (FTPA).

Frequent T-Patterns Algorithm (FTPA)
Build occurrence table; *K*=2; Find couple T-Patterns with their significant CIs; While(termination condition){ Find *K*-Tuple T-patterns form (*k-1*)-Tuple T-Patterns set; *k= k+1*; } Return the set of T-patterns with their CIs;

Figure 9-17. Frequent T-Patterns algorithm pseudo code.

9.3.3.6. Experimental Results on FTPA

To evaluate FTPA performance in finding frequent activity we used the 3 resident pilot dataset where the bathroom visits were labeled. We note that labeling is not necessary in our MFA framework: if the activity is frequent, we don't need to know its name.

First, we run our FTPA algorithm on a training sensor dataset to extract all T-Patterns of size k ($2 \leq k \leq d$, where there is no significant CI for pattern of size d+1) with their significant CIs and save them in a dictionary of T-Patterns. Then for a given test sensor sequence, we apply a top-down approach such that we find the longest T-Pattern first that matches with one of T-patterns in the dictionary. If the interval of pattern of size *k* does not belong to the CI from the dictionary, then we look for T-Pattern of size (*k*-1), where *k*>=3 (see Figure 9-18).

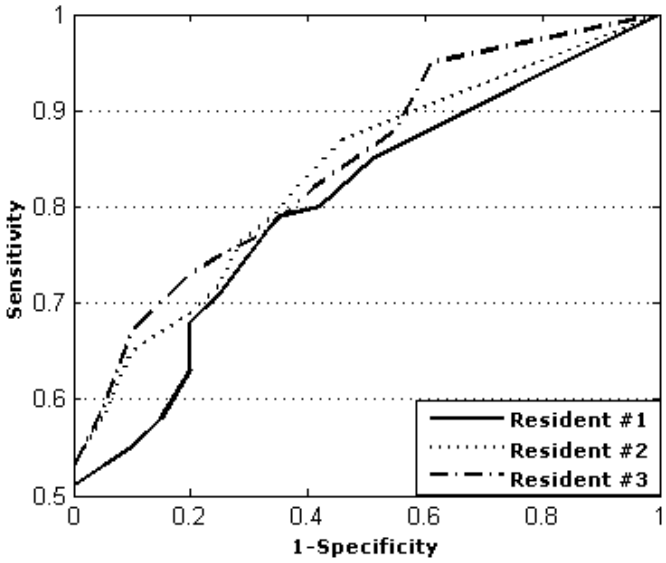
Frequent pattern detection using FTPA
Step1. Build dictionary of T-Patterns using FTPA and sensor data. *k*=*d*; Step2. If *k*>=2, then look for pattern of size *k* in the given sensor sequence such that their CI matches. Step 3. If nothing found, then *k*=*k*-1, and repeat Step2.

Figure 9-18. Finding frequent patterns using FTPA algorithm.

To evaluate the performance of the proposed method we use *sensitivity* and *specificity* as defined before. Table 9-6 presents the result of activity recognition using a 10 cross-fold validation approach. Figure 9-19 shows the results in a ROC curve as well as AUC separately for each of the residents. The average recognition performance is 0.83. Resident #2 and #3 use a walker that affects their speed reflected in the sensor hits. Therefore, the extraction process has a lower sensitivity, compared to Resident #1.

Table 9-6. Frequent pattern detection perfromance

Resident ID	**Sensitivity**	**Specificity**
#1	0.85	0.76
#2	0.83	0.69
#3	0.83	0.71



AUC #1	AUC #2	AUC #3
0.80	0.83	0.84

Figure 9-19. Performance of frequent pattern detection using FTPA based on AUC.

Resident #1 is the youngest and most active, which allows him to have a less rigid routine, with bathroom visits that are more varied in time. Also he is the only one taking a diuretic medication which in turn may affect the number of bathroom visits. Both Residents #2 and #3 use walkers that may force them to have more regimented and planned bathroom trips.

9.4. Motif Discovery in Bioinformatics

In bioinformatics, the problem of identifying and characterizing shared subsequences, motifs, in a set of unaligned protein sequences are called motif discovery. A motif is simply a common pattern between a set of nucleic or amino acids sequences that share some biological features or properties. In computer science terminology, motif discovery is the problem of identifying a set of non-overlapping, approximately matching substrings given a set of strings. In a simpler version of this problem, given a dataset of biopolymer sequences which are known to contain a single shared motif, the problem is to find in each sequence the starting position of the appearance of the motif and to describe the shared motif. In a more general problem there might be multiple motifs in

the given set of strings. This is still an open question, since there is no assumption regarding the width, position, letter frequencies, or even the number of the motifs that may exists in the given dataset. In the rest of this chapter, we apply and test a bioinformatics motif detection approach on TigerPlace dataset and discuss the results.

9.4.1. Unsupervised Learning of Multiple Motifs

Motif discovery, the problem of finding multiple, distinct shared motifs in a set of biological sequences, has been studied by researches extensively. In 1990, Hertz et al. [221] introduced a greedy algorithm for discovering a single, shared motif in a given set of sequences. Bailey et al. [222] used EM algorithm and introduced MEME (Multiple EM for Motif Elicitation) algorithm to identify motifs in unaligned biopolymer sequences where little or nothing is known in advance about any motifs that may be present in the given dataset. Three main ideas of MEME algorithm are [222]:

1. To increase the likelihood of finding globally optimal motifs, the MEME algorithm initializes the EM starting point with subsequences that actually occur in the given dataset. Therefore, EM iteratively converges to the locally optimum motifs while increases the likelihood of converging to the global optimal motifs.
2. A heuristic modification of EM is applied in such a way that relaxes the assumption that each sequence contains exactly one occurrence of the shared motif. Therefore, multiple variations of a motif can occur in any sequence. In this way, the algorithm can ignore sequences with no occurrence of the shared motif. This increases the resistant of the algorithm to the noisy data.
3. After motifs are found, they will be probabilistically erased. Therefore, several motifs can be identified within the same set of sequences.

One can consider MEME algorithm as a learning tool while other motif discovery methods such as BLAST or FASTA are searching tools. In a searching tool a pattern is searched within a given dataset of sequences. The pattern can be another sequence, a consensus subsequence, regular expression defining a motif, or a more high-level combination of features. However, in a learning tool a supervised learning method

discovers motifs (patterns) that all sequences share in a given set of sequences. In an unsupervised motif discovery approach a set of sequences are given. The unsupervised learning tool discovers motifs that some of sequences share. So the unsupervised learning algorithm must simultaneously look for a cluster of input sequences and a pattern that the members of this cluster do have in common. MEME algorithm performs as an unsupervised method.

9.4.2. MEME Algorithm

MEME algorithm modifies the EM-based method proposed in [219] for motif discovery. The EM algorithm finds a probabilistic model of shared motifs in a given unaligned sequences while a motif length (W) is known. The idea of this method is that there is one occurrence of the motif in each sequence in the dataset. However, the position of the motif in each sequence is unknown. This model has been referred to as the "one-occurrence-per-sequence" model or "one-per" model in [220]. In this model, each example of the motif is assumed to have been generated by a sequence of independent, discrete random variables. Therefore, the observed frequencies of letters in the columns are the maximum likelihood estimations of the distributions of the variables. Since the offset of the motifs are unknown, they should be estimated as well. The EM algorithm here comes to play and estimates the probability that the shared motif starts in position j of sequence i in the dataset. These probability estimates, z_{ij}, are then used to re-estimate the probability of letter l in column c of the motif, ρ_{lc} , for each letter in the alphabet where $1 \leq c \leq W$. The EM algorithm uses Bayes' rule to re-estimates z and ρ until ρ changes slightly from one iteration to the other. Figure 9.20 shows the pseudo-code of EM algorithm.

```
EM (dataset, W){
        Choose starting point (ρ)
        do{
            re-estimate z from ρ
            re-estimate ρ from z
        } until (change in ρ <ϵ )
        return
}
```

Figure 9-20. Pseudo-code of EM algortihm.

In EM algorithm, z refers to the matrix of offset probabilities z_{ij}, and ρ refers to the matrix of letter probabilities ρ_{ij}. In an iterative process, the EM algorithm finds the model of the motifs as the sequence of independent discrete random variables with parameter ρ and estimates z matrix, the probability of each possible starting point of examples of the motif in the sequences in the dataset. The likelihood of the model given a training dataset is the probability of the dataset given the model. The EM algorithm finds the values of the parameters of the model such that it maximizes the expected likelihood of the generating data given the model ρ and matrix z. Readers are referred to [220] for more details about re-estimating matrix z and matrix ρ. The logarithm of the likelihood for the "one-per" model is calculated using Eq. 9.6.

$$\log(likelihood) = N\sum_{j=1}^{W}\sum_{l\in\mathcal{L}} f_{lj}\log(\rho_{lj}) + N(L-W)\sum_{l\in\mathcal{L}} f_{l0}\log(\rho_{l0}) + Nlog(\frac{1}{L-W+1}) \quad (9.6)$$

In Eq. 9.6, N is the number of sequences in the dataset, L is the length of the sequences, W is the length of the shared motif, $\mathcal{L}$ is the alphabet of the sequences, ρ_{lj} (unknown) is the probability of letter l in position j of the motif, ρ_{l0} (unknown) is the probability of letter l in all non-motif positions, f_{lj} is the observed frequency of the letter l in position j of the motif, and f_{l0} is the observed frequency of the letter l in all non-motif positions of the sequences [220]. The EM algorithm finds the values of the parameters of the model such that it maximizes the likelihood function locally. However, it is desirable to find values of the parameters of the model that maximized the likelihood function globally. The EM method and the one-per model have several limitations that are pointed bellow.

9.4.3. Limitations of EM Method:

1. Finding the best values of the starting point (initial values) for the matrix ρ is difficult. Because it affects the evaluation of the motifs and when we decide that the correct shared motif is discovered.

2. The one-per model assumes that there is exactly one occurrence of the shared motif in each sequence of the dataset. Therefore, sequences with multiple occurrences of the shared motif will have less contribution to the model while sequences with no occurrences of the shared motif will over-contribute to the characterization of the shared motif. Because in the one-per model approach having more sequences with no occurrences of the shared motif makes it impossible for the EM to find the shared motif.
3. In the one-per model, the EM algorithm assumes that there is exactly one shared motif in the sequences; so after characterizing one shared motif, it does not look for other motifs. Therefore the EM algorithm with one-per model approach cannot find multiple motifs or motifs with insertions of variable lengths.

The MEME algorithm extends the EM algorithm with one-per model approach to address the limitations mentioned above. The MEME algorithm considers all sequences in the dataset to systematically initialize the starting points. It also allows the system to choose one-per model or *n*-per model that allows each sequence to have zero or more (up to *n*) occurrences of the shared motif. Also, the MEME algorithm probabilistically erases the appearances of a motif after it is found and searches for other motifs. Figure 9.21 shows the pseudo code of MEME algorithm.

```
MEME algorithm (dataset, W, NSITES, PASSES){
    for i=1 to PASSES{
        for each subsequence in dataset{
            run EM algorithm 1 iteration with starting point derived from this subsequence
            choose model of shared motif with highest likelihood
            run EM to converge from starting point which generate that model
            print converged model of shared motif
            erase appearances of shared motif from dataset
        }
    }
}
```

Figure 9-21. MEME algorithm.

In the MEME algorithm, *PASSES* is the number of different motifs, *NSITES* is the number of occurrences of a shared motif in the dataset. If *NSITES* is set equal to the number of sequences in the dataset, then it is possible for the *n*-per model to get approximately the same results as the one-per model on a dataset that has one appearance of the shared motif in each sequence. When *NSITES* is less than the number of sequences in the dataset, MEME can assign very low offset probabilities to all positions in a sequence that does not contain the motif at all. Therefore, it helps to reduce the problem of sequences which do not contain the motif blurring its characteristics. The exact value of *NSITES* is not critical for the discovered motifs by MEME algorithm. So it is not necessary to know in advance exactly how many times a motif is presented in the dataset. The MEME algorithm uses a modified version of EM algorithm in an inner loop and repeatedly evaluates different models with different starting points (either one-per model or *n*-per model). Different starting points are selected from actual sequences that are occurred in the dataset. Then the EM algorithm runs just one time per each starting point and produces a probabilistic model of the possible shared motifs. The probabilistic model with highest likelihood will be chosen as the starting point of the EM model to run over to converge. Therefore, the model of the shared motif is discovered. Then, all occurrences of the shared motif in the dataset are erased and in the outer loop the system repeats the whole process to discover more shared motifs. In the next section, we run the MEME tool [221] on TigerPlace dataset to find activity patterns and compare those results with FTPA algorithm that we proposed earlier in this chapter.

9.4.4 Experimental Results Using MEME Method

The MEME Suite [222] is a software toolkit with a unified web server interface that has four types of motif analysis including motif discovery, motif–motif database searching, motif-sequence database searching and assignment of function. The MEME Suite toolkit includes MEME and GLAM2 for motif discovery, TOMTOM for searching similar motifs in databases of known motifs, FIMO, GLAM2SCAN and MAST for searching for occurrences of motifs in sequence databases, and GOMO for finding associations between motifs and GO terms. The components of the MEME Suite are

implemented in ANSI C as command line tools published as SOAP (Simple Object Access Protocol) web services using Opal and the Tomcat Java servlet container.

9.4.4.1. Experiments Setup

Using MEME tool to find motif, there are some parameter initializations as described in the following.

- *Motif Distribution*: this parameter specifies the user information (prior belief) about the distribution of the occurrences of the motifs in the sequence dataset. Initializing the correct distribution for this parameter improves the sensitivity and quality of the motif discovery process. The motif distribution can be selected as: *one per sequence; zero or one per sequence; and any number of repetitions*. By choosing the first option, MEME assumes that each sequence in the dataset contains exactly one occurrences of each motif, as we referred to as one-per model before. The second option assumes that each sequence may contain at most one occurrence of the motif. This is helpful when we suspect that some motifs might be missing from some of the sequences. In this case, this option will provide more accurate results compare to the first distribution option. The last distribution option assumes any number of non-overlapping occurrences of a motif in the sequences in the database. This is useful when we suspect that motifs may repeat multiple times within a single sequence in the dataset. This option takes more computational times (about ten times as much) and is less sensitive to weak motifs which do not repeat within a single sequence than the other two options.
- *Number of motifs*: this is the maximum number of motifs that MEME will search for in the training set. MEME will stop when this number of motifs has been found, or when none can be found with E-value less than 10000.

- *Number of sites*: this is the total number of sites (places) in the training dataset where each motif has occurred. The minimum and the maximum number of occurrences can be selected in case there is a prior knowledge about the number of occurrences which limits MEME's search and increases the likelihood of finding true motifs. MEME may still find motifs with slightly fewer or more occurrences then those you specify. These fields are optional. If they left blank, MEME will choose limits depending on the type of the selected occurrence distribution and the number of sequences (*n*) in the training dataset using Table 9-7.

Table 9-7. Default Numbers of Sites for each Motif.

Type of distribution	**minimum sites**	**maximum sites**
one occurrence per sequence	*n*	*n*
zero or one occurrence per sequence	*sqrt(n)*	*n*
any number of repetitions per sequence	*sqrt(n)*	*min(5*n, 50)*

- *Motif width:* this is the number of characters in the motif, aka motif length. MEME uses a statistical heuristic function to choose the optimal length of each motif individually. However, different limits for the maximum and the minimum of the motif width can be chosen.
- And there are more sophisticated options for finding motifs in protein sequences using MEME that we ignore as they are not relevant to the topic of this dissertation.

Table 9-8 summarizes the parameter initialization for this experiment on three residents of TigerPlace.

Table 9-8. MEME parameter initialization.

Resident#	***Motif Distribution***	***Number of Motifs***	***Number of Sites***	***Motif width***
#1	Any number of repetitions	10	Default	6-50
#2	Any number of repetitions	10	Default	6-50
#3	Any number of repetitions	10	Default	6-50

We used the dataset described in Table 9-4. For simplicity in sensor motifs representation, we map each sensor to a letter as shown in Table 9-9.

Table 9-9. Mapping sensors to letters for MEME algorithm.

Letter	Sensors
A	Bedroom motion
F	Bedroom: Bed motion
C	Bed sensors: BedMovement1, BedMovement2, BedMovement3, BedMovement4
D	Bed sensors: Breathing1, Breathin2, Breathing3
E	Bed sensors: Pulse1, Pulse2, Pulse3
K	Kitchen sensors: Cabinet, Cup Cabinet, Drawer, Plate Cabinet, Refrigerator, Silverware Drawer, Motion
W	Kitchen sensors: Temp High, Temp Low
M	Living room motion
T	Bathroom motion
L	Laundry room motion
S	Shower motion
G	Closet motion
N	Medicine cabinet motion
I	Office motion
R	Dining room motion
V	Vest room motion
H	Chair: off, on
Q	Den motion

We used the online MEME tool that takes the sequences in a FASTA format. Therefore, we map each sensor to a letter to represent the sensor log file as a FASTA file where for the purpose of motif detection we remove the time stamp. Figure 9-22 shows an example of sensor log file prepared as the input for MEME online tool.

Figure 9-22. Formated sensor log file to be used in MEME online tool for motif detection.

9.4.4.2. Results and Discussions

We extracted and analyzed the frequent patterns of three TigerPlace residents using MEME online tool [225] over a period of 10 days as we represent in the following. Moreover, we performed the same set of experiments using FTPA method (see section 9.3.3) and compared the results with MEME which we represent at the end of this section.

Resident #1

This resident is a 99 years old male who uses walker and has urinary problem. Apparently in the period of time that we used for experiment his bed sensors did not worked! Therefore, there is no bed sensor firing for this resident. Table 9-10 shows motifs extracted using MEME tool for this resident where ***#of sites*** indicates the number of occurrences of the motif.

Table 9-10. Extracted motifs for resident #1 using MEME online tool

Motif	*# of Sites*	*Sequence*	*Activity*
1	10		Bathroom visit at day time
2	10		Bathroom visit at mid night
3	10		Kitchen activities during mid night
4	10		Invloves motion (perhaps walks) from bedroom to living room and to the kitchen. Usually happened at night time.
5	10		Shorter walks happened mostly in day

			time.
6	10		Walk through bedroom and closet and bathroom

MEME patterns have been manually verified and their descriptions are provided in the third column of Table 9-10. There are two major categories of frequent activities (motifs), walks and daily activities such as food preparation that are referred to as kitchen activities, or bathroom visits.

Resident #1 has frequent midnight bathroom visits (motif #2). Mid night bathroom visits are different than day time bathroom visits (motif #1) in patterns (the order of sensor events), lengths and durations. Usually day time bathroom visits involve more shower sensor hits that coincidence with shower visits.

Motif #3 represents kitchen activities during the night that involves bedroom motion sensor hits. This motif might be an indication of dinner preparation or midnight cravings.

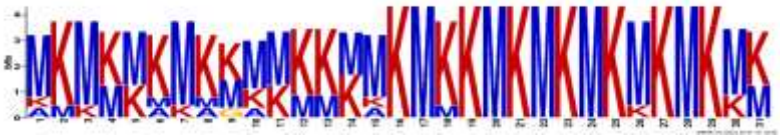
Motifs #4, #5, and #6 represent walks in the apartment. While motif #4 shows a longer walk through living room, kitchen, and bedroom, motif #5 shows a shorter walk usually happened during the day time that involves mostly bedroom and closet.

Resident #2

Resident #2 is an 88 years old male with no urinary problem. He does not use walker. However, he does take diuretic medication which affects his bathroom visits. Apparently, in the period of time that we used for experiment his bed sensors did not worked! Therefore, there is no bed sensor firing for this resident either. Results have been shown in Table 9-11.

Table 9-11. Extracted motifs for resident #2 using MEME online tool

Motif	# of Sites	Sequence	Activity
1	3		Shower visit
2	10		Morning bathroom visit
3	10		Evening bathroom visits
4	10		Bathroom visits during day time
5	10		Short bathroom visits during day time
6	5		Another walk from bathroom to bedroom, living room at night time

7	4		Night time kitchen activity
8	8		Day time kitchen activity

There are two major categories of frequent activities (motifs), walks and daily activities such as food preparation that are referred to as kitchen activities, or bathroom visits.

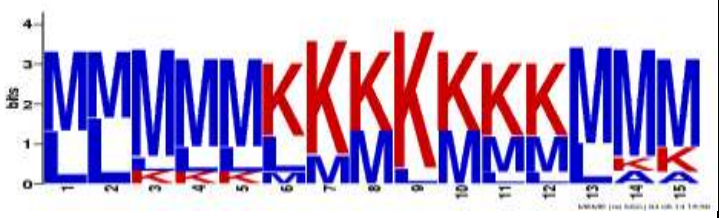
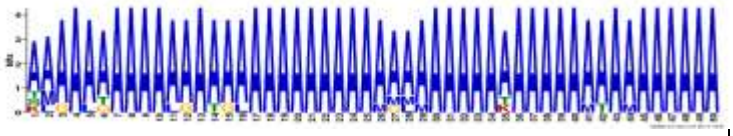
Resident #2 has frequent shower visits (motif #1) that happened 3 times during the period of 10 days of this study. Motif #2 shows morning bathroom visits that are short visits. Motif #3 shows longer night time bathroom visits. However, bathroom visits of day time vary in length as it is shown in motifs #4 and #5.

Motif #6 represents walks in the apartment through bathroom, bedroom, and living room. Motif #7 and #8 represent kitchen activities during the night and day time respectively. Night time kitchen activities that are shorter in length may correspond to snacks preparation or midnight cravings.

Resident # 3

Resident #3 is a 90 years old female with urinary problems. She uses walker. However, she does not take diuretic medication. Apparently, in the period of time that we used for experiment his bed sensors did not worked! Therefore, there is no bed sensor firing for this resident.

Table 9-12. Extracted motifs for resident #3 using MEME online tool

Motif	***# of Sites***	***Sequence***	***Activity***
1	10		Day time walks through laundry room to living room and kitchen
2	10		Bathroom visit in the evenings time
3	6		Afternoon activities in the living room and laundry room
4	8		Bathroom visit happened at day time
5	10		Bedroom activity at night time
6	6		afternoon or evening time walks through closet, bedroom, living room, den, and kithen

Since resident #3 uses walker, she has more rigid activities. More specifically, her bathroom visits are more consistent compare to bathroom visits of resident #1 and

resident #2 that have been shown in table 9-10 and table 9-11 respectively. Motif #2 and motif #4 shows her bathroom visits both frequently happen in the evening and day time, respectively. Resident #3 has no frequent mid night bathroom trips according to results of MEME online tool.

Motif #1 and motif #6 shows her frequent trips (referred to as walks in the table). Motif #1 shows her activities, perhaps a walk, through living room, laundry room, and kitchen happening usually in the day time, while motif #6 pictures a trip through closet, bedroom, living room, den (or kitchen) frequently happening in the afternoons or evenings.

Motif #5 shows her significantly frequent and long activity in the bedroom at nights. Moreover, motif #3 shows that she has a frequent afternoon activity wandering through Den room, living room, kitchen, bedroom, closet, and in some cases laundry room where her laundry room is actually in the bathroom area.

Comparing FTPA and MEME Results:

We run the same experiment using FTPA method on the sensor data of three residents over the same period of 10 days to extract motifs. To compare FTPA method with MEME method, we focused on the bathroom visits, because we can verify results more accurately with less human effort. Figure 9-23 shows bathroom patterns that are identified by MEME and FTPA methods in a period of 10 days for resident #1 of TigerPlace. To extract frequent bathroom patterns, we manually extracted sensor sequences of bathroom visits of resident #1. FTPA performs directly on the sensor log files while MEME needs formatted sensor log file as described before.

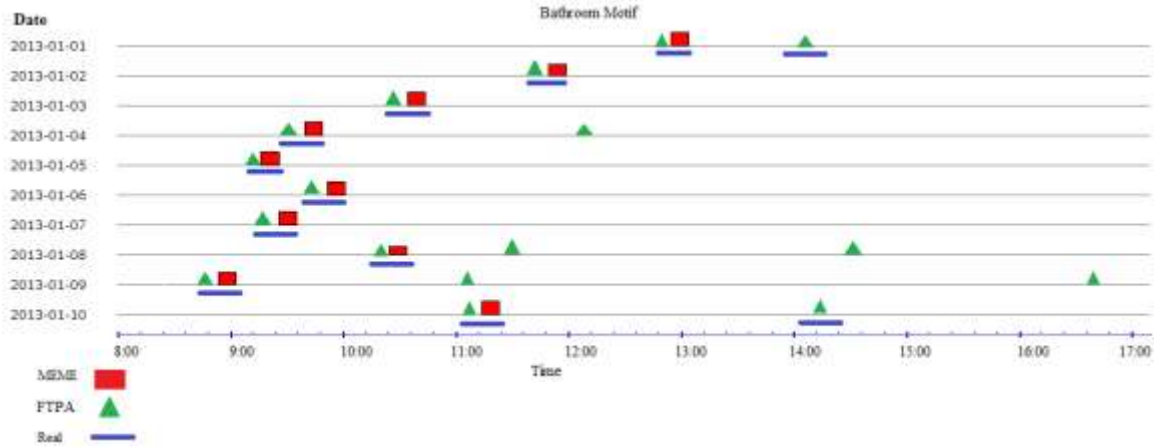


Figure 9-23. Frequent bathroom patterns of resident #1 identified by MEME and FTPA methods

As we see in Figure 9-23, both methods discover bathroom visits very well. However, FTPA algorithm finds afternoon bathroom visits more accurately. The reason is afternoon bathroom visits are shorter visits in terms of the number of sensor events during the visit and they might not happen in the exact same time each day. However, MEME algorithm finds frequent patterns based on their location in the sequence. Therefore, afternoon bathroom visits that happen at different time in the afternoon are not identified as frequent patterns by MEME algorithm. While FTPA method identifies frequent visits based on the order of sensor events and the length of activity. Therefore, even if the bathroom visits happen in different times of afternoon, as long as the order of the sensor events are almost similar and the critical interval is statistically significant (i.e. the duration of visit is almost similar), the bathroom visit is identified as the frequent bathroom visit by FTPA method. However, FTPA has false alarms that are identifying some activities as bathroom visit which are not really a bathroom visit. This is because

most of them (false alarms) are short activities in the closet that are identified frequent by FTPA.

We evaluate the behavior of MEME and FTPA methods on general activities of the residents over the same period of 10 days as well, plotted in Figure 9-24.

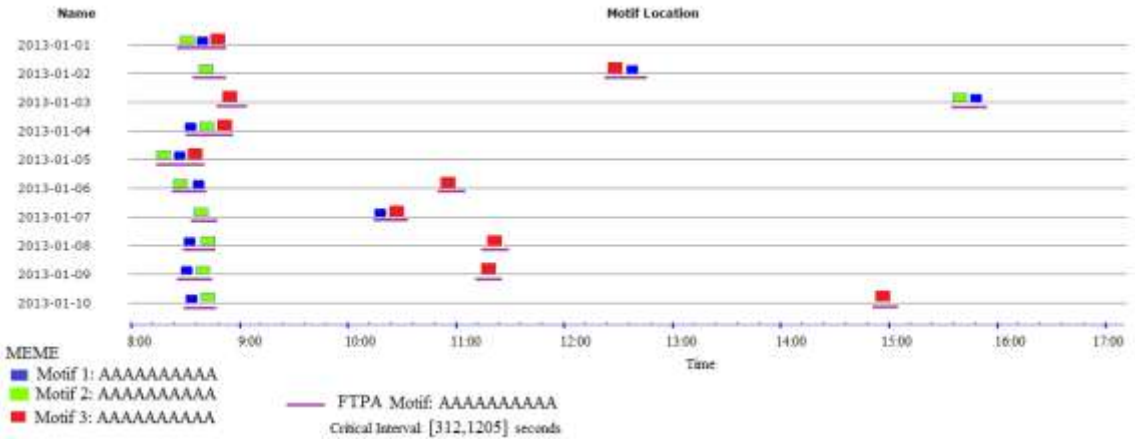


Figure 9-24. Frequent activties of resident #1 identified by MEME and FTPA methods

Figure 9-24 shows three frequent activities (motifs) found by MEME method where the maximum size of patterns are set to 10. The figure shows patterns related to spending time in the bedroom. Three patterns are exactly the same in terms of activities but different in terms of the time of day they happened. However, FTPA method identifies them as one pattern with a statistically significant critical interval. Figure 9-25 shows the section of sensor file related to these patterns. This experiment shows the effect of parameter initialization on MEME method. If the length of motifs are not known before the hand, and parameters are not initialized properly, MEME method may results poorly and may splits a single motif into several smaller motifs, as we see in Figure 9-25.

Date	Motif 1	Motif 2		Motif 3
2013-01-10	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA
2013-01-09	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA
2013-01-08	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA
2013-01-07	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA
2013-01-06	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA
2013-01-05	AGGTGTTGGG	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA
2013-01-04	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA
2013-01-03	MKKMKMKMKM	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA
2013-01-02	MAKKMKKKAM	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA
2013-01-01	AAAAAAAAAA	AAAAAAAAAA	AAAAAAAAAA	GAAAAAAAAA

Figure 9-25. snap shot of sensor log files related to three patterns shown in Figure 9-24.

While MEME distinguishes between same activities that happen in different times, FTPA finds them similar as long as the order of sensor events are similar and the whole activity completes in a statistically significant critical interval. As it is shown in Figure 9-25, three patterns are pretty much similar except they happen in different times. The advantages of FTPA in defining three pattern as one single frequent pattern is in the time it saves while searching for missing frequent activities. Since the same activities are represented by a single frequent pattern, the search for missing frequent activity can finds one much faster than frequent activities that are identifies by MEME. Figure 9-26 shows the number of similar patterns that are identified by MEME and FTPA based on the length of patterns.

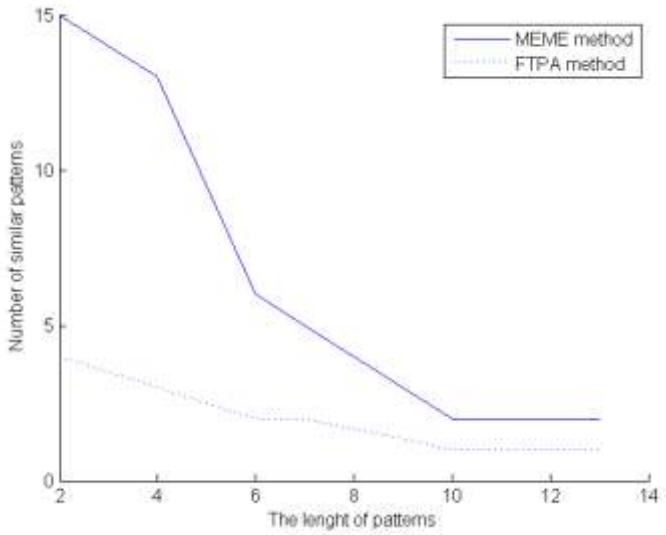


Figure 9-26. The number of similar patterns of resident #1 identified by MEME and FTPA method based on the length of patterns

In Figure 9-26 solid line shows that MEME method distinguishes between similar patterns with shorter patterns compare to longer patterns. That is because, shorter patterns such as patterns of size 2 (i.e. two sensor events) are happen more frequently than larger patterns. Therefore, there are more variations in their location. As a result, MEME method defines them as different patterns. As the length of patterns increases, since there are fewer occurrences of them, the MEME algorithm forms them well. However, for mid size patterns such as pattern of size 4 or 6, the number of similar patterns that are identified as different patterns by MEME is still high. This may increase the search time and the false alarm rate.

In Figure 9-26 dotted line shows that FTPA performs better in terms of grouping similar patterns. The way FTPA distinguishes between similar frequent patterns is with the definition of the critical interval. For example, one frequent bathroom visit can be shown as, Pattern 1: (*GTTTG*), with a critical interval, CI=[180,300] seconds, that means

the whole activity end between 180 second to 300 seconds. Another frequent bathroom visit can be defined as, Pattern 2: (*GTTTG*), with a critical interval, CI=[600,900] seconds, that means the whole activity end between 600 second to 900 seconds. Therefore, when searching for missing frequent activity, the monitoring system searches for a single pattern (*GTTTG*), if the pattern has occurred, depending on the duration of the pattern, the system selects that either Pattern 1 or Pattern 2 has happened. This way, the system saves huge amount of time in searching compared to the MEME frequent patterns.

9.5. Conclusion

We described a new early illness recognition framework, MFA, based on frequent pattern analysis. Since MFA is based on a frequent pattern analysis method, we explored a modified version of the T-Pattern algorithm, FTP as a possible candidate. On a pilot dataset, our modified T-Pattern algorithm had an average recognition rate of bathroom visits of about 0.83 with about 0.3 false alarm rate, which makes it a reasonable candidate for our MFA framework.

However, the complete implementation of the MFA framework will be performed in future work. Aside of FTP algorithm, it will include the activity similarity measure developed in [14] together with a relational clustering technique.

Chapter 10 Conclusion and Future Directions

In this dissertation, we used sensor network technology to detect changes in health status of elderly living alone, alert health care providers, and augment traditional health care. To this aim, we addressed several major challenges including:

- We discussed the problem of measuring the temporal similarity of two multi-dimensional time series and proposed a novel similarity measure for MATS data.
- We proposed novel approaches to predict health patterns using time series data collected from sensor networks.
- We proposed a novel framework to annotate sensor sequences data using contextual health information of residents of smart homes.
- We designed efficient machine learning approaches for motif detection on MATS data.

To measure the similarity of multi-dimensional time series, we focused on health care applications. We reviewed already discovered similarity functions for time series. Inspiring from bioinformatics approach, we proposed a new method to measure the similarity of two multi-dimensional time series. We introduced the idea of modified Smith-Waterman framework, Temporal Smith-Waterman, for early illness recognition that uses temporal sequences. We analyzed the properties of TSW, introduced a faster way to compute it based on genetic programming, and proposed a new method for searching large time series. Our approach doesn't require series data conversion to continuous format. Instead, we arranged all different sensor hits together with their time stamps into a one-dimensional sequence. Our method overcomes difficulties related to data uncertainty and aggregation that often arise when processing sensor data.

For health pattern recognition, we described a new frame work that uses TSW for pattern recognition and its applications to eldercare and early illness recognition. We described a framework for predicting abnormal health patterns using non-wearable sensor sequence similarity. A sensor pattern is classified as "abnormal" if it is much smaller than the mean of the distribution of "normal" patterns similarities. "Abnormal" days are

defined by unusual sensor activity patterns that require a nurse's assessment of the resident.

Moreover, to annotate the sensor sequences we improved the proposed framework such that it detects a change in resident's health status and predicts the possible health issues based on sensor data and nursing notes produced by an integrated in-home monitoring system. We proposed a novel semi-supervised clustering method that automatically uses user provided information to effectively cluster nursing notes. This framework represents each cluster by a language model. We used the bi-grams model of terms to consider term to term dependencies and improve the efficiency of the proposed clustering method. We estimated the cluster model by a maximum a posterior probability to address the challenge of highly imbalance nursing visit notes dataset.

We also proposed three methods to identify deviations in patterns of day-to-day activities of older adults to generate alerts to the healthcare providers for timely interventions. Daily routines, such as bathroom visits, can be monitored by automated in-home sensor systems. We presented novel approaches that find periodicity in sensor time series data using clustering approach, item set mining approach, and statistical approach. A retrospective multiple case study (N=3) design was used to quantify bathroom visits as parts of the older adult's daily routine, over a 10-day period.

Moreover, we discussed the problem of finding frequent sensor patterns or motifs in sensor time series. We reviewed existing techniques to mine periodic behaviors from sensor time series data, with a focus on addressing difficulties raised in real applications such as multiple interleaving periods, partial time span, and noises or outliers. We presented new approaches based on frequent item set mining and statistical approach. These approaches are used to observe residents activities and detect periodicity from the sensor sequences. The periodic sensor sequences are proposes to annotate sensor sequences and predict any abnormality in resident's health status.

10.1. Future Directions

We acknowledge that the proposed approaches have limitations. We discuss these limitations here and outline future research directions.

1. TSW measures the similarity of two sensor sequences. If the resident history of activities (as sensor sequences) is long and the monitoring system has to perform the search for many (say thousands) residents simultaneously, this method can be very slow. We proposed a genetic algorithm approach to overcome this challenge. However, there are other approaches that may further improve the search efficiency such as Indexing approaches and lower boundary searches.

2. Predicting abnormal events is the main aim of this work. We proposed two approaches to address this problem. However, we used labeled data for that. Applying unsupervised approach is a challenging task that can be investigated in the future researches.

3. Predicting the future health status of residents is another future direction. To this aim, the first step is to recognize the recent health status of residents. In this dissertation, we proposed several approaches to detect and recognize the recent health status of residents using motif discovery. Researches can utilize these methods and build a predictive model based upon them to predict the future health status and their health trend over a period of time.

4. Finding the correlation between motion data collected from the sensor networks and the health status collected from the nursing notes is still a challenging problem. In this dissertation we proposed a semi-supervised approach to address this problem. Other unsupervised approaches can be investigated in this regard.

5. In this dissertation we focused on personalized health monitoring systems. However, there are still open questions in this regards such as: Are residents with the same disease follow the same routines every day? How do we compare routines of different residents? What metrics are more indicative of the health status of elderly resident? And so many more questions that can be researched.

References

[1] G. M. Pogorelc B, "Home-based health monitoring on the elderly through gait recognition," *Journal of Ambient Intelligence and Smart Environments,* pp. 415-428, 2012.

[2] H. B. Z. B. K. Y.-F. Goodwin JS, "Risk of Continued Institutionalization after Hospitalization in Older Adults," the Journal of Gerontology, Medical Science, pp. 1321-7, 2011.

[3] Editor, "TigerPlace: An Innovative Aging in Place Community," The American Journal of Nursing, pp. 68-69, 2013.

[4] A. V. KV Grayson, "The next four decades: the older population in the United States: 2010 to 2050," U.S. Department of Commerce, Economics and Statistics Administration, 2010.

[5] M. P. J. K. TL Hayes, "An unobtrusive in-home monitoring system for detection of key motor changes preceding cognitive decline," in Proc. of the 26th Annual Intl. Conf. of the IEEE EMBS, San Francisco, CA, 2004.

[6] L. K. M.-T. E. B. J. K. P. N. J. e. a. Intille S, "Using a live-in laboratory for ubiquitous computing research," in Proc. of Pervasive Health, 2006.

[7] O. R. A. G. A. I. E. B. M. B. e. a. Kidd CD, "The Aware Home: A living laboratory for ubiquitous computing research," in Proc of the 2nd International Workshop on Cooperative Buildings-CoBuild'9, 1999.

[8] K. L. H. G. Haigh KZ, "Independent Lifestyle Assistant: Lessons learned," Journal of Assistive Technology, pp. 87-106, 2006.

[9] A. G. P. M. R. M. K. J. Skubic M, "A Smart Home Application to Eldercare: Current Status and Lessons Learned.," Journal of Technology and Health Care, pp. 183-201, 2009.

[10] S. M. Heise D, "Monitoring pulse and respiration with a non-invasive hydraulic bed sensor," in Proc 32nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Buenos Aires, Argentina, 2010.

[11] B. R., "Designing for ubiquity: The perception of privacy," Journal of Pervasive Computing, pp. 40-46, 2003.

[12] A. M. T. B. S. R. F. R. Mack D, "passive and portable system for monitoring heart rate and detecting sleep apnea and arousals: Preliminary validation," in Proceedings Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare (D2H2), Arlington, VA, 2006.

[13] D.-M. K. A. M. T. H. S. M. D. G. e. a. Rantz MJ, "technology and nursing collaboration to help older adults," Nursing Outlook, pp. 40-45, 2005.

[14] W. M. Smith TF, "Identification of common molecular subsequences," Journal of Molecular Biology, pp. 195-197, 1981.

[15] M. V. D. E. L. M. G. M. Kaluza B, "An Agent-based Approach to Care in Independent Living," Ambient Intelligence, Lecture Notes in Computer Science, pp. 177-186, 2010.

[16] C. f. D. C. a. P. (CDC)., "Health Places Termonology," 20 August 2012. [Online].

[17] M. K. A. M. e. a. Rantz MJ, "A technology and nursing collaboration to help older adults age in place," Nurs Outlook, pp. 40-45, 2005.

[18] S. M. M. S. G. C. A. G. K. J. P. M. Rantz MJ, "Sensor Technology to Support Aging in Place," JAMDA- Journal of AMDA, pp. 386-391, 2013.

[19] P. J. S. P. e. a. Mack DC, "Development and preliminary validation of heart rate and breathing rate detection using a passive, ballistocardiographybased sleep monitoring system," IEEE Trans Inf Technol Biomed, pp. 111-120, 2009.

[20] M. R. a. Y. M. C. Faloutsos, "Fast subsequence matching in time-series databases," in Proceedings of the 1994 ACM SIGMOD international conference on Management of data, 1994.

[21] C. F. a. A. S. R. Agrawal, "Efficient similarity search in sequence databases," Foundations of Data Organization and Algorithms, pp. 69-84, 1993.

[22] K.-P. C. a. A.-C. Fu, "Efficient time series matching by wavelets," in 15th International Conference on Data Engineering (ICDE), Sydney, 1999.

[23] T. K. a. A. Singh, "Variable length queries for time series data," in 17th International Conference on Data Engineering (ICDE), Heidelberg, Germany, 2001.

[24] X. T. a. W. Z. C. Shahabi, "TSA-tree: A wavelet-based approach to improve the efficiency of multi-level surprise and trend queries," in 12th International Conference on Scientific and Statistical Database Management (SSDBM) , Berlin, Germany, 2000.

[25] T. H. a. J. Bezdek, "An efficient formulation of the improved visual assessment of tendency (iVAT) algorithm," IEEE Trans. Knowledge and Data Engineering, pp. 813-822, 2005.

[26] P.-N. Tan, M. Steinbach and V. Kumar, Introduction to Data Mining, Addison-Wesley, 2005.

[27] M. R. E. H. G.A. Ten Holt, "Multi-Dimensional Dynamic Time Warping for Gesture Recognition," in Thirteenth annual conference of the Advanced School for Computing and Imaging, 2007.

[28] M. Muller, Information Retreival for musin and motion, Springer, 2007.

[29] M. Popescu, "An ontological fuzzy Smith-Waterman with application to patient retrieval in Electronic Medical Records," in IEEE International Conference on Fuzzy Systems (FUZZ), 2010.

[30] A. S. Karlin S, "Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes," Proc. Natl. Acad. Sci. USA, pp. 2264-2268, 1990.

[31] C. G. O. R. S. M. R. M. Popescu M, "An Eldercare Electronic Health Record System for Predictive Health Assessment," in IEEE International Conference on Health Communication 2011, Columbia, MO, 2011.

[32] M. Skubic, "ElderTech," [Online]. Available: https://eldertech.missouri.edu/overview.htm.

[33] E. M. J Rowan, "Digital Family Portrait field trial: support for Aging in Place," in Proc. of the SIGCHI Conf. on Human Factors in Computing Systems, New York, USA, 2005.

[34] J. W. C. G. M. G. P. Cuddihy, "Algorithm to automatically detect abnormally long periods of inactivity in a home," in Proc. of the 1st ACM SIGMOBILE Intl. Workshop, New York, USA, 2007.

[35] B. T. C. E. S. S. S. M. &. P. M. Rantz M, "Automated Fall Detection with Quality Improvement "Rewind" to Reduce Falls in Hospital Rooms," Journal of Gerontological Nursing Technology Innovations, vol. 40(1), pp. 13-17, 2014.

[36] A. M. T. B. S. R. F. R. Mack D, "A passive and portable system for monitoring heart rate and detecting sleep apnea and arousals: Preliminary validation," in Proceedings Transdisciplinary Conference on Proceedings Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare (D2H2). , Arlington VA., 2003.

[37] T. C. Fu, "A review on time series data mining," Engineering Applications of Artificial Intelligence , vol. 24, no. 1, pp. 164-181, 2011.

[38] M. P. E. Keogh, "A simple dimensionality reduction technique for fast similarity search in large time series databases," Knowledge Discovery and Data Mining, Current Issues and New Applications, vol. 1805, pp. 122-133, 2000.

[39] H. T. Buu and D. T. Anh, "Time Series Discord Discovery Based on iSAX Symbolic Representation," in Third International Conference on Knowledge and Systems Engineering (KSE), 2011 .

[40] Z. Yang, "Application of symbolic techniques in detecting determinism in time series," in Proceedings of the 20th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp.2670–2673, 1998.

[41] M. Motoyoshi, "Mining temporal classes from time series data," in Proceedings of the 11th ACM International Conference on Information and Knowledge Management, pp.493–498, 2002.

[42] W. Aref, "Incremental, online, and merge mining of partial periodic patterns in time-series databases," IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 3, p. 332–342, 2004.

[43] J. Lin, "A Symbolic representation of timeseries,with implications for streaming algorithms," in Proceedings of 8th ACM SIGMOD International Conference on Management of Data Workshop on research Issues in Data Mining and Knowledge Discovery, pp.2–11, 2003.

[44] J. a. Lin, "Experiencing SAX: a novel symbolic representation of timeseries," Data Mining and Knowledge Discovery, vol. 15, no. 2, p. 107–144, 2007.

[45] V. Megalooikonomou, "A Dimensionality reduction technique forefficientsimilarityanalysisoftimeseriesdatabases," in 13th ACM International

Conference on Information and Knowledge Management, pp.160–161, 2004.

[46] F. Morchen, "Optimizingtimeseriesdiscretizationforknowledge discovery," in 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pp.660–665, 2005.

[47] M. Magnuson, "Discovering hidden time patterns in behavior: T-pattern and their detection," Behavior Research Methods, vol. 32, pp. 93-110, 2000.

[48] A. P. E. e. a. Salah, "T-Pattern Revisited: Mining for Temporal Patterns in Sensor Data," Sensors, pp. 7496-7513, 2010.

[49] G. Salton, The SMART retrieval system: Experiments in automatic document processing, USA: Prentice-Hall, 1971.

[50] P. R. a. H. S. C. Manning, Introduction to Information Retrieval, USA: Cambridge Univ. Press, 2008.

[51] M. M. U. A. K. e. a. H Wang, "Evaluation of local spatio-temporal features for action recognition," in In Proceeding of BMVC, London, 2009.

[52] L. M. P. P. e. a. M. Chen, "Exploiting multi-level parallelism for low-latency activity recognition in streaming video," in ACM SIGMM Conf. on Multimedia Systems, 2010.

[53] A. K. C. S. a. C. L. H. Wang, "Action recognition by dense trajectories," in Proceeding of CVPR, 2011.

[54] W. Z. S. Y. a. A. N. Q. Le, "Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis," in Proceeding of CVPR, 2011.

[55] S. M. A. J. e. a. R. Hamid, "A novel sequence representation for unsupervised analysis of human activities," Artificial Intelligence, vol. 173, no. 14, pp. 1221-1244, 2009.

[56] K. K. Karwarth A, "Relational Sequence Alignments and Logos," in Proceedings of the 16th International Conference on Inductive Logic Programming (ILP-06), Spain, 2006.

[57] M. B. S. K. Mrozek D, "EAST: Energy Alignment Search Tool," Fuzzy Systems and Knowledge Discovery, pp. 696-705, 2006.

[58] E. &. K. S. Keogh, "On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration," in the 8 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Alberta, Canada, 2002.

[59] M. A. D. I. P. &. M. R. Gavrilov, "Mining the stock market: which measure is best?," in the 6th ACM Int'l Conference on Knowledge Discovery and Data Mining, Boston, MA, 2000.

[60] X. &. S. P. Ge, "Deformable markov model templates for time-series pattern matching," in the 6th ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining, Boston, MA, 2000.

[61] K. B. &. F. E. Pratt, "Search for patterns in compressed time series," Int'l Journal of Image and Graphics, vol. 2, no. 1, p. 89–106, 2002.

[62] K. F. a. B. Kedem, "Regression Theory for Categorical Time Series," Statistical Science, vol. 18, no. 3, p. 357–376, 2003.

[63] P. a. N. J. A. MCCULLAGH, Generalized Linear Model, London: Chapman and Hall, London, 1989.

[64] L. a. K. H. FAHRMEIR, "Regression models for nonstationary categorical time series.," Journal of Time Series Analysis, p. 147–160, 1987.

[65] H. KAUFMANN, "Regression models for nonstationary categorical time series: Asymptotic estimation theory," Ann.Statistics, vol. 15, p. 79–98, 1987.

[66] K. K. B. a. S. D. FOKIANOS, "Predicting Precipitation Level," J. Geophys. Res. D: Atmospheres, p. 473–26, 1996.

[67] D. R. M. P. A. I. R. A. e. a. BRILLINGER, "Some wavelet-based analyses of Markov Chain Data," Signal Processing, vol. 80, p. 1607–1627, 2000.

[68] Agresti, Categorical Data Analysis,, Wiley, 2002.

[69] C. D. R., "Partial likelihoodBiometrika," Biometrika, vol. 62, p. 269–276, 1975.

[70] H. M., Martingale, Encyclopedia of Mathematics, Springer, 2001.

[71] Y. D. G. e. a. Chen, "Multi-Dimensional Regression Analysis of Time-Series Data Streams," in Proceeding of the 28th VLDB Conference, China, 2002.

[72] H. Kaufmann, "Regression models for nonstationary categorical time series: asymptotic estimation theory," Annual Statistics, vol. 15, p. 79–98, 1987.

[73] T. F. K. Moysiadis, "On binary and categorical time series models with feedback," Journal of Multivariate Analysis, vol. 131, p. 209–228, 2014.

[74] T. W. R.M.Jong, "Dynamic time series binary choice," Econometric Theory, vol. 27, p. 673–702, 2011.

[75] S. Ahmad, "Summarization of multimodal information," in Proceedings of the Fourth International Conference on Language Resourcesand Evaluation , pp.1049–1052, 2004.

[76] A. Guttman, "R-trees:adynamicindexstructureforspatialsearching," in ACM SIGMOD International Conference on Management of Data, pp.47–57, 1984.

[77] R. Agrawal, "Efficient similarity search in sequence databases," in The Fourth International Conference on Foundations of Data Organization and Algorithms, pp.69–84, 1993.

[78] C. Faloutsos, "Fastsubsequencematchingin time-seriesdatabases," in ACM SIGMOD International Conference on Management of Data, pp.419–429, 1994.

[79] K. Yang, "A Multi level distance-based index structure for multivariate timeseries," in 12th IEEE International Symposium onTemporal Representation and Reasoning, pp.65–73, 2005.

[80] M. Vlachos, "A Multi-metric index for Euclidean and periodic matching," in The Ninth European Conference on Principles and Practice of Knowledge Discovery in Databases, pp. 355–367, 2005.

[81] M. Vlachos, "Indexing multi-dimensional time-series with support for multiple distance measures," in the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.216–225, 2003.

[82] J. Shieh, "iSAX: disk-aware mining and indexing of massivetimeseries datasets," Data Mining and Knowledge Discovery, vol. 19, no. 1, p. 24–57, 2009.

[83] P. A. C. Esling, "Time-Series Data Mining," ACM Computing Surveys (CSUR), vol. 45, no. 1, pp. 1-12, 2012.

[84] C. R. M. e. a. Faloutsos, "Fast subsequence matching in time-series databases," in SIGMOD, pp. 419–429, 1994.

[85] H. T. G. S. P. W. X. A. K. E. DING, "Querying and mining of time series data: Experimental comparison of representations and distance measures," in VLDB Endowm, pp. 1542–1552, 2008.

[86] K. K. W. C. a. M. H. Wong, "Fast time-series searching with scaling and shifting," in the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS), pp. 237-248 , 1999.

[87] E. J. Keogh, "Efficiently finding arbitrarily scaled patterns in massive time series databases," in PKDD, pp. 253–265, 2003.

[88] V. A. W. R. CHHIENG, "Adaptive distance measurement for time series databases," Lecture Notes in Computer Science, Springer, vol. 4443, p. 598–610, 2010.

[89] V. W. Q. L. G. A. F. C. MEGALOOIKONOMOU, "A multiresolution symbolic representation of time series," in 21st International Conference on Data Engineering, pp. 668–679, 2005.

[90] J. A. L. Y. LIN, "Finding structural similarity in time series data using bag-of-patterns representation," in 21st International Conference on Scientific and Statistical Database Management, pp. 461–477, 2009.

[91] Y. A. Y. D. XIONG, "Time series clustering with ARMA mixtures," Pattern Recognition, vol. 37, no. 8, pp. 1675-1689, 2004.

[92] M. F. C. A. M. G. DEGLI ESPOSTI, "Sequence distance via parsing complexity: Heartbeat signals," Chaos, Sol. Fractals, vol. 39, no. 3, p. 991–999, 2009.

[93] J. G. A. CHAPPELIER, "A Kohonen map for temporal sequences," in Conference on Neural Networks and Their Applications, pp. 104–110, 1996.

[94] P. SMYTH, "Clustering sequences with hidden Markov models," Adv. Neural Info. Process. Syst., p. 648–654, 1997.

[95] H. Y. K. A. S. C. YOON, "Feature subset selection and feature ranking for multivariate time series," IEEE Trans. Knowl. Data Engin., p. 1186–1198, 2005.

[96] G. H. B. HEBRAIL, "Symbolic representation of long time-series," in Symbolic Data Analysis at the 4th European Conference on Principles of Data Mining and Knowledge Discovery, pp. 56–65.

[97] E. J. T. KEOGH, "Clustering of time series subsequences is meaningless: Implications for previous and future research," in 3rd IEEE International Conference on Data Mining, pp. 115–122, 2003.

[98] P. K. E. L. J. A. L. S. PATEL, "Mining motifs in massive time series databases," in IEEE International Conference on Data Mining (ICDM02), pp. 370–377, 2002.

[99] A. DENTON, "Kernel-Density-Based clustering of time series subsequences using a continuous random-walk noise model," in the 5th IEEE International Conference on Data Mining, pp. 122–129, 2005.

[100] B. S. G. BAKSHI, "Representation of process trends–IV. Induction of real-time patterns from operating data for diagnosis and supervisory control," Comput. Chemi. Engin., vol. 18, no. 4, p. 303–332, 1994.

[101] E. A. P. M. KEOGH, "An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback," in the 4th International Conference of Knowledge Discovery and Data Mining, pp. 239–241, 1998.

[102] P. GEURTS, "Pattern extraction for time series classification," in the 5th European Conference on Principles of Data Mining and Knowledge Discovery, pp. 115–127, 2001.

[103] S. H. Y. JENG, "Time series classification based on spectral analysis," Comm. Statisti. Simul.Comput., vol. 37, no. 1, p. 132–142, 2008.

[104] J. K. L. RODRIGUEZ, "Time series classification: Decision forests and SVM on interval and DTW features," in the Workshop on Time Series Classification, 13th International Conference on Knowledge Discovery and Data Mining, 2007.

[105] X. K. E. S. C. W. L. R. C. XI, "Fast time series classification using numerosity reduction," in 23rd International Conference on Machine Learning, 2006.

[106] D. R. C. SRISAI, "Efficient time series classification under template matching using time warping alignment," in the 4th International Conference on Computer

Sciences and Convergence Information Technology, pp 685–690, 2009.

[107] T. K. N. A. B.-J. Z. LIN, "Alignment and classification of time series gene expression in clinical studies," Bioinformatics, vol. 24, no. 13, pp. 147-155, 2008.

[108] T. E. M. M. W. A. H. B. LOWITZ, "Hidden markov models for classification of heart rate variability in RR time series," in World Congress on Medical Physics and Biomedical Engineering, pp. 1980–1983, 2009.

[109] A. A. R. A. M. Y. NANOPOULOS, "Feature-Based classification of time-series data," Information Processing and Technology, pp. 49-61, 2001.

[110] R. J. M. L. A. A. Y. J. POVINELLI, "Time series classification using Gaussian mixture models of reconstructed phase spaces," IEEE Trans. Knowl. Data Engin., vol. 16, no. 6, pp. 779-783, 2004.

[111] A. SUBASI, "EEG signal classification using wavelet feature extraction and a mixture of expert model," Expert Syst. Appl., vol. 32, no. 4, pp. 1084-1093, 2007.

[112] C. A. W. D. RATANAMAHATANA, "Stopping criterion selection for efficient semi-supervised time series classification," Studies Comput. Intell., vol. 149, pp. 1-14, 2008.

[113] X. W. J. Y. X. O. H. A. L. T. ZHANG, "A novel pattern extraction method for time series classification," Optimiz. Engin., vol. 10, no. 2, pp. 253-271, 2009.

[114] F. C. M. L. A. L. F. A. A. B. LOTTE, "A review of classification algorithms for EEG-based brain–computer interfaces," J. Neural Engin., vol. 4, pp. 1-13, 2007.

[115] E. K. S. L. Bill Chiu, "Probabilistic Discovery of Time Series Motifs," in KDD '03 Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003.

[116] P. A. P. S. C. A. B. R. FERREIRA, "Mining approximate motifs in time series," Lecture Notes in Computer Science, vol. 4265, p. 89–101, 2006.

[117] D. K. E. J. C. B. A. Z. V. YANKOV, "Detecting time series motifs under uniform scaling," in 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 844–853, 2007.

[118] Y. A. N. T. MOHAMMAD, "Constrained motif discovery in time series," New Gener. Comput., vol. 27, no. 4, p. 319–346, 2009.

[119] E. J. K. S. L. S. E. Thanawin Rakthanmanon, "MDL-based time series clustering," Knowledge and Information Systems, vol. 33, no. 2, pp. 371-399, 2012.

[120] Y. L. J. O. T. Li, "Visualizing Variable-Length Time Series Motifs," in SIAM International Conference on Data Mining, pp. 895-906, 2012.

[121] R. E. J.-M. O. Jagannadan Varadarajan, "A Sequential Topic Model for Mining Recurrent Activities from Long Term Video Logs," International Journal of Computer Vision, vol. 103, no. 1, pp. 100-126, 2013.

[122] P. Geurts, "Patternextractionfortimeseriesclassification," in Proceedings of the Fifth European Conference on Principles and Practice of Knowledge Discovery in Databases, pp. 115–127, 2001.

[123] H. Zhang, "A Non-parametric wavelet feature extractor for time-series classification," in Proceedings of the Eighth Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp.595–603, 2004.

[124] M. Kadous, "Classification of multi variate time series and structured data using constructive induction," Machine Learning, vol. 58, p. 179–216, 2005.

[125] K. Yang, "CLeVer: a feature subset selection technique for multivariate timeseries," in Proceedings of the Ninth Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp.516–522, 2005.

[126] R. Povinelli, "Time series classification using Gaussian mixture models o freconstructed phase spaces," IEEE Transaction on Knowledge and Data Engineering, vol. 16, no. 6, p. 779–783, 2004.

[127] F. E. Popescu M, "Linking Clinical Events in Elderly to In-home Monitoring Sensor Data: A Brief Review and a Pilot Study on Predicting Pulse Pressure," Journal of Computing Science and Engineering, pp. 180-199, 2008.

[128] M. A. Popescu M, "Early Illness Recognition in Older Adults Using In-Home Monitoring Sensors and Multiple Instance Learning," Methods of Informatics in Medicine, pp. 359-367, 2012.

[129] H. D. P. M. Lee Y, "Microphone Array System for Automatic Fall Detection," IEEE Transactions on Biomedical Engineering, pp. 1291-1301, 2012.

[130] A. S. A. G. T. Kahveci, "Similarity Searching for Multi-attribute Sequences," SSDBM, pp. 175-184, 2002.

[131] H. M. G. D. K. E. Vlachos M, "Indexing Multi-Dimensional Time-Series with Support for Multi Distance Measures," SIGKDD, pp. 216-225, 2003.

[132] S. P., "Multiple Multidimensional Sequence Alignment Using Generalized Dynamic Time Warping.," WSEAS Transaction on Mathematics. , pp. 684-694, 2012.

[133] V. A. P. V. P.-S. X. e. a. Bautista MA, "Probability-based Dynamic Time Warping for Gesture Recognition on RGB-D data," Advances in Depth Image Analysis and Applications, pp. 126-135, 2013.

[134] P. M., "An ontological fuzzy Smith-Waterman with application to patient retrieval in Electronic Medical Records.," in IEEE International Conference on Fuzzy Systems (FUZZ), 2010.

[135] C. G. O. R. S. M. R. M. Popescu M, "An Eldercare Electronic Health Record System for Predictive Health Assessment.," in IEEE 13th International Conference on e-Health Networking, Applications and Services. , 2011.

[136] S. V., "Using Pervasive computing to deliver elder care.," IEEE Pervasive Computing, pp. 10-13, 2002.

[137] V. U., "Pervasive healthcare and wireless health monitoring," Mobile Network and Applications, pp. 113-127, 2007.

[138] R. S. S. P. Pallikonda M, "Elderly patient monitoring system using wireless sensor network," Telemedicine and e-Health, pp. 73-79, 2009.

[139] C. D. S. F. W. M. Reinert G, "Alignment-Free Sequence Comparison (I): Statistics and Power," Journal of Computational Biology, pp. 1615-1634, 2009.

[140] P. M. Hajihashemi Z, "Predicting Health Patterns Using Sensor Sequence Similarity and NLP," in 2012 IEEE International Conference on BIBM, Philadelphia, USA, 2012.

[141] G. C. R. V. Duchene F, "Learning recurrent behaviors from heterogeneous multivariate time-series," Journal of Art. Intell. Med., pp. 25-47, 2007.

[142] M. G. F. K. R. DAVIS, "Physical activity patterns assessed by accelerometry in older people," European Journal of Applied Physiology, pp. 581-589, 2007.

[143] G. Z. L. B. W. J. R. M. T. S. N. J. YANG, "From sensor network to behavior profiling: a homecare perspective of intelligent buildings.," in IEEE Seminar for Intelligent Buildings, 2004.

[144] N. M. H. N. A. A. J. L. H. J. C. R. G. A. T. e. a. GIL, "Data visualization and data mining technology for supporting care for older people," in Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility, New York, 2007.

[145] D. Y. J. M. R. W. H. D. CHEN, "Detecting social interactions of the elderly in a nursing home environment," in ACM Transactions on Multimedia Computing, Communications and Applications, 2007.

[146] A. G. D. H. A. WILLIAMS, "Aging in place: fall detection and localization in a distributed smart camera network," in Proceedings of the 15th International Conference on Multimedia, New York, 2007.

[147] M. D. S. M. D. K. S. T. B. L. e. a. ALWAN, "Impact of monitoring technology in assisted living: outcome pilot.," in IEEE Transaction on Information Technology in BioMedicine, 2006.

[148] N. P. A. EAGLE, "Reality mining: sensing complex social systems," Personal and Ubiquitous Computing, pp. 255-268, 2006.

[149] J. A. C. H. S. E. FOGARTY, "Sensing from the basement: a feasibility study of unobtrusive and low-cost home activity recognition," in In Proceedings of the 19th Annual ACM Symposium on User interface Software and Technology., New York, 2006.

[150] A. J. N. SIXSMITH, "A smart sensor to detect the falls of the elderly," IEEE Pervasive Computing, pp. 42-47, 2004.

[151] E. M. I. S. S. L. K. TAPIA, "Activity recognition in the home using simple and ubiquitous sensors," Proceedings of the Second International Conference on Pervasive Computing. Springer, pp. 158-175, 2004.

[152] J. S. C. C. S. W. X. K. P. Y. SHIEH, "Remote monitoring of mobility changes of the elderly at home using frequency rank order statistics.," Journal of Medical and Biological Engineering, pp. 81-88, 2006.

[153] G. N. N. D. J. VIRONE, "A system for automatic measurement of circadian activity deviations in telemedicine," IEEE Transactions on Biomedical Engineering, pp. 1463-1469, 2002.

[154] G. W. A. S. L. C. Q. F. L. D. T. e. a. VIRONE, "An assisted living oriented information system based on a residential wireless sensor network.," in the 1st Distributed Diagnosis and Home Healthcare (D2H2) Conference, Arlington, VA, USA, 2006.

[155] T. H. N. S. D. W. A. Bar-Hillel, "Learning distance functions using equivalence relations," in the 12th International Conference on Machine Learning, 2003.

[156] A. B. R. M. S. Basu, "Semi-supervised clustering by seeding," in Proceedings of the 19th International conference on Machine Learning, 2002.

[157] A. B. R. M. S. Basu, "Active semi-supervision for pairwise constrained clustering," in Proceedings of the SIAM International Conference on Data Mining, 2004.

[158] W. C. J. Pone, "A language modeling approach to information retrieval," in Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1998.

[159] W. L. Ruizhang Huanga, "An active learning framework for semi-supervised document clustering with language modeling," Data & Knowledge Engineering, pp. 49-67, 2009.

[160] Z.-J. L. Chou-Yuan Lee, "A novel algorithm applied to classify unbalanced data," Applied Soft Computing, p. 2481–2485, 2012.

[161] I. B. M.A.H. Farquad, "Preprocessing unbalanced data using support vector machine," Decision Support Systems, p. 226–233, 2012.

[162] O. C. F. Y. Loïs Rigouste, "Inference and evaluation of the multinomial mixture model for text clustering," Information Processing & Management, p. 1260–1280, 2007.

[163] G. K. V. K. M. Steinback, "A comparison of document clustering techniques," in KDD Workshop on Text Mining, 2000.

[164] G. K. Y. Zhao, "Hierarchical clustering algorithms for document datasets," Data Mining and Knowledge Discovery, pp. 141-168, 2005.

[165] X. H. J. H. D. Cai, "Document clustering using locality preserving indexing," IEEE Transactions on Knowledge and Data Engineering, p. 1624–1637, 2005.

[166] S. C. J. H. Y. Li, "Text document clustering based on frequent word meaning sequences," Data and Knowledge Engineering, pp. 381-404, 2008.

[167] Z. Z. W. L. R. Huang, "Text clustering with limited user feedback under local metric learning," in Proceedings of Asia Information Retrieval Symposium (AIRS),, 2006.

[168] S. Zhong, "Semi-supervised model-based document clustering: a comparative study," Machine Learning, pp. 3-29, 2006.

[169] C. C. K. Wagstaff, "Clustering with instance-level constraints,," in Proceedings of the 17th International Conference on Machine Learning, 2000.

[170] S. K. C. M. D. Klein, "From instance-level constraints to space-level constraints: making the most of prior knowledge in data clustering," in Proceedings of the 19th International Conference on Machine Learning, 2002.

[171] D. Y. H. Chang, "Locally linear metric adaptation for semi-supervised clustering and image retrieval," Pattern Recognition, pp. 1253-1264, 2006.

[172] R. C. A. M. D. Cohn, "Semi-supervised clustering with user feedback," Cornell University, Technical Report TR2003-1892, 2003.

[173] S. R. I. Davidson, "Clustering with constraints: feasibility issues and the k-means algorithm," in Proceedings of the SIAM International Conference on Data Mining, 2005.

[174] K. K. D. P. N. Kumar, "Semi-supervised clustering with metric learning using relative comparisons," in Proceedings of 5th IEEE International Conference on Data Mining, 2005.

[175] A. N. M. J. S. R. E.P. Xing, "Distancemetric learning, with application to clustering with side-information," Advances in NIPS, p. 521–528, 2003.

[176] T.-H. C. H.-H. L. Chien-Liang Liu, "Clustering documents with labeled and unlabeled documents using fuzzy semi-Kmeans," Fuzzy Sets and Systems, pp. 48-64, 2013.

[177] L. C. W.-C. T. Yang Yan, "Fuzzy semi-supervised co-clustering for text documents," Fuzzy Sets and Systems, pp. 74-89, 2013.

[178] Z. Z. W. L. R. Huang, "Text clustering with limited user feedback under local metric learning," in Proceedings of Asia Information Retrieval Symposium (AIRS),, 2006.

[179] B. L. R. G. e. a. Yang C., "Incremental and Decremental Affinity Propagation for Semisupervised Clustering in Multispectral Images," IEEE Transactions on Geoscience and Remote Sensing, pp. 1666 - 1679, 2013.

[180] S. P. ,. S. L. ,. F. W. e. a. Yangqiu Song, "Constrained Text Coclustering with Supervised and Unsupervised Constraints," IEEE Transactions on Knowledge and Data Engineering,, pp. 1227 - 1239, 2013.

[181] K.-I. Fukui, S. Ono, T. Megano and M. Numao, "Evolutionary Distance Metric Learning Approach to Semi-supervised Clustering with Neighbor Relations," IEEE 25th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 398 - 403, 2013.

[182] C.-O. D. Asafi S., "Constraints as Features," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013.

[183] D. T. Y. R. J. C. Jun Yua, "Pairwise constraints based multiview features fusion for scene classification," Pattern Recognition, p. 483–496, 2013.

[184] P. W. S. C. Hao Xia, "Online multi-modal distance learning for scalable multimedia retrieval," in Proceedings of the sixth ACM international conference on Web search and data mining(WSDM '13), 2013.

[185] S. L. e. a. Xiting Wang, "Mining evolutionary multi-branch trees from text streams," in KDD '13 Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, 2013.

[186] C. D. B. Yan, "Subspace metric ensembles for semi-supervised clustering of high dimensional data," in 509–520, 2006.

[187] Z. Hajihashemi, "An Early Illness Recognition Framework Using a Temporal Smith Waterman algorithm and NLP," in AMIA Annu Symp Proc, 2013.

[188] J. L. C. Zhai, "A study of smoothing methods for language models applied to information retrieval," ACM Transactions On Information Systems, p. 179–214, 2004.

[189] J. L. A. Berger, "Information retrieval as statistical translation," in Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval,

222-229.

[190] B. W. Guodong Ding, "GJM-2: A Special Case of General Jelinek-Mercer Smoothing Method for Language Modeling Approach to Ad Hoc IR," in the Second Asia Information Retrieval Symposium (AIRS2005), 2005.

[191] H. B. D. Kok, "Natural Language Processing for the working programmer," 2010.

[192] J. J. M. P. V. O. e. a. Guergana K Savova, " Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications," JAMIA, pp. 507-513, 2009.

[193] T. Joachims, "A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization," Computer Science Technical Report CMU-CS-96-118, Carnegie Mellon University, 1996.

[194] M. Wurst, "The Word Vector Tool," University of Dortmund, Department of Computer Science, 2006.

[195] J. H. Zhenhui Li, "Mining Periodicity from Dynamic and Incomplete Spatiotemporal Data," Data Mining and Knowledge Discovery for Big Data, pp. 41-81, 2014.

[196] N. M. a. D. W. C. Huiping Cao, "Discovery of periodic patterns in spatiotemporal sequences," IEEE Transactions on Knowledge and Data Engineering, pp. 453-467, 2007.

[197] H. C. G. K. M. H. Y. T. a. D. C. N. Mamoulis, "Mining, indexing, and querying historical spatiotemporal data," in Proc. 2004 ACM SIGKDD Int. Conf.Knowledge Discovery in Databases (KDD’04), Seattle, WA, 2004.

[198] Y. T. M. A. a. S. P. Yuni Xia, "Reducing data redundancy in location-based services," GeoSensor, 2006.

[199] Q. L. H. T. S. a. X. Z. Hoyoung Jeung, "A hybrid prediction model for moving objects," in Proc. 2008 Int. Conf. Data Engineering (ICDE’08), Cancun, 2008.

[200] E. K. ,. S. L. ,. P. P. Jessica Lin, "Finding surprising patterns in a time series database in linear time and space.," in SIGKDD ’02, Alberta, Canada, 2002.

[201] X. &. S. P. Ge, "Deformable Markov model templates for time-series pattern matching," in proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, 2000.

[202] K. G. D. &. P. V. Kalpakis, "Distance measures for effective clustering of ARIMA time-series," in proceedings of the 2001 IEEE International Conference on Data Mining, San Jose, CA,, 2001.

[203] E. C. K. P. M. &. M. Keogh, "Dimensionality reduction for fast similarity search in large time series databases," Journal of Knowledge and Information Systems, pp. 263-286, 2000.

[204] E. C. K. P. M. &. M. S. Keogh, "Locally adaptive dimensionality reduction for indexing large time series databases," in proceedings of ACM SIGMOD Conference on Management of Data., Santa Barbara, CA, 2001.

[205] P. A. &. S. S. H. Pevzner, "Combinatorial approaches to finding subtle signals in DNA sequences," in proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology, La Jolla, CA, 2000.

[206] M. &. B. J. Tompa, "Finding motifs using random projections.," in proceedings of the 5th Int'l Conference on Computational Molecular Biology, Montreal, Canada, 2001.

[207] G. L. K. M. H. R. G. &. S. P. Das, "Rule discovery from time series," in the 4th Int'l Conference on Knowledge Discovery and Data Mining, New York, NY, 1998.

[208] F. Höppner, "Discovery of temporal patterns -- learning rules about the qualitative behavior of time series.," in Proceedings of the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases, Freiburg, Germany, 2001.

[209] J. Y. P. W. W. a. H. J. Yang, "Mining long sequential patterns in a noisy environment.," in proceedings SIGMOD International Conference on Management of Data, 2002.

[210] Y. J. a. Y. P. Wang. W., "Meta-patterns: revealing hidden periodical patterns," in Proceedings of the 1st IEEE International Conference on Data Mining, 2001.

[211] P. K. N. &. M. S. Indyk, "Identifying representative trends in massive time series data sets using sketches," in In proceedings of the 26th Int'l Conference on Very Large Data Bases, Cairo, Egypt, 2000.

[212] D. H. R. e. A. Amy McGovern, "Identifying predictive multi-dimensional time series motifs: an application to severe weather prediction," Data Min Knowl Disc, p. 232–258, 2011.

[213] K. E. L. W. L. S. Lin J, "Experiencing SAX: a novel symbolic representation of time series," Data Min Knowl Discov, p. 107–144, 2007.

[214] S. T. a. J. M. P. Avrilia Floratou, "Efficient and Accurate Discovery of Patterns in Sequence Data Sets," IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, pp. 1154-1168, 2011.

[215] E. K. Abdullah Mueen, "Online Discovery and Maintenance of Time Series Motifs," in KDD '10 Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, 2010.

[216] J. Hipp, U. Güntzer and G. Nakhaeizadeh, "Algorithms for association rule mining - a general survey and comparison," ACM SIGKDD Explorations , pp. 58-64 , 2000.

[217] R. S. F. Baraglia, " Dynamic personalization of web sites without user intervention," Communication of the ACM, pp. 63-67, 2007.

[218] D. C. J. J. e. A. Barbara, "ADAM: Detecting Intrusions by Data Mining," in Proceedings of the IEEE Workshop on Information Assurance and Security, 2001.

[219] R. S. Rakesh Agrawal, " Fast algorithms for mining association rules in large databases," in Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, 1994.

Vita

Zahra was born in Isfahan, Iran in 1982. She finished her high school at 2000 with a diploma in mathematics and physics. She then started her undergrad degree in computer software engineering. She got her bachelor degree in 2005. Then, she moved to Tehran, the capital of Iran, for her master's degree in 2006. She got her master's degree at artificial intelligence and robotics from the Iran University of Science and Technology (IUST) at 2008. She went back to her home town, Isfahan, and was an instructor in three colleges and worked as data scientist part time for two years.

To pursue her dream and achieving the highest level of education, she moved to the US in 2010 to attend the PhD program in Computer Science department of the University of Missouri at Columbia, MO, (MU). She finished her course works within two years and started her research under Dr. Mihail Popescu supervision in 2012. She published 7 conference papers as the first author, 2 as second author, as well as 2 journal papers. Finally, she successfully defended her PhD dissertation in March 10, 2015.