12-18-2013

A Framework for Discovery and Diagnosis of Behavioral Transitions in Event-streams

Arash Akhlaghi
Georgia State University

A FRAMEWORK FOR DISCOVERY AND DIAGNOSIS OF BEHAVIORAL TRANSITIONS IN EVENT-STREAMS

by

ARASH AKHLAGHI

Under the Direction of Dr. William Robinson

ABSTRACT

Date stream mining techniques can be used in tracking user behaviors as they attempt to achieve their goals. Quality metrics over stream-mined models identify potential changes in user goal attainment. When the quality of some data mined models varies significantly from nearby models—as defined by quality metrics—then the user's behavior is automatically flagged as a potentially significant behavioral change. Decision tree, sequence pattern and Hidden Markov modeling being used in this study. These three types of modeling can expose different aspect of user’s behavior. In case of decision tree modeling, the specific changes in user behavior can automatically characterized by differencing the data-mined decision-tree models. The sequence pattern modeling can shed light on how the user changes his sequence of actions and Hidden Markov modeling can identifies the learning transition points. This research describes how model-quality monitoring and these three types of modeling as a generic framework can aid recognition and diagnoses of behavioral changes in a case study of cognitive rehabilitation via emailing. The date stream mining techniques mentioned are used to monitor patient goals as part of a clinical plan to aid cognitive rehabilitation. In this context, real time data mining aids clinicians in tracking user behaviors as they attempt to achieve their goals. This generic framework can be widely applicable to other real-time data-intensive analysis problems. In order to illustrate this fact, the similar Hidden Markov modeling is being used for analyzing the transactional behavior of a telecommunication company for fraud detection. Fraud similarly can be considered as a potentially significant transaction behavioral change.

INDEX WORDS: Patient monitoring, Behavioral rehabilitation, Behavioral transition, Clinical treatment plans, Cognitive rehabilitation, Monitoring, Learning, Stream mining, Real-time data mining, Decision trees, Sequence pattern, Max motif, Hidden Markov model, Fraud detection, Transaction processing

A FRAMEWORK FOR DISCOVERY AND DIAGNOSIS OF BEHAVIORAL TRANSITIONS IN EVENT-STREAMS

by

ARASH AKHLAGHI

A Dissertation Submitted in Partial Fulfillment of the Requirements for Degree of

Doctor of Philosophy

In the College of Arts and Sciences

Georgia State University

2013

A FRAMEWORK FOR DISCOVERY AND DIAGNOSIS OF BEHAVIORAL TRANSITIONS IN EVENT-STREAMS

by

ARASH AKHLAGHI

DEDICATION

To my parents that always supported me in every aspect of my life.

ACKNOWLEDGEMENT

I would like to thank my research advisor Dr. William Robinson. He had a crucial contribution to this study and I learned a lot from him regarding research methodology, academic technicalities and writing skills. There were times that we hit some serious road blocks in our study but his brain storming, patience, our regular meetings and discussions opened some new ventures that eventually ended up to some interesting and practical discoveries and resulted to publications of several conference and journal papers. I also want to thank him for his financial support for several years being a PhD student at Georgia State University.

My special thanks go to Dr. Raj Sunderraman my program advisor. He admitted me to the PhD program and helped me continuously during my studying. His guidance during my studying led me to where I stand today. He also energized me by his wonderful courses offered in past two semesters and I learned a lot from him. He also gave me the opportunity teaching in CSc. department and helped me to grow in that domain too.

Next, I would like to thank my committee members, Dr. Anu Bourgeois and Dr. Xiaolin Hu for their valuable suggestions during my dissertation proposal in improving the quality of this work.

I would also like to thank one of the research team members, Tanjie Deng, who helped me in making progress in our study and took care of some of the tasks and coding of our research.

And the last but not the least, I want to thank CIS department especially Dr. Carl Stucke that gave me the opportunity for several years to teach programming and application development which are my fields of passion and gaining invaluable experience in teaching and interaction with different type of students.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

The world is filled with events. Birthdays, loan applications, weather changes, ticket purchases, *etc*. are among the many events in modern life. Such events can be digitized and analyzed. Tibco Software was founded to facilitate an organized response to events placed on the information bus. Tibco, IBM, and many others are part of the trend to rearchitect enterprise software from its variations on client-server to an information bus, reactive system. Converging trends of the information bus ($16.4 billion 2010), business intelligence and analytics ($13.1 billion 2012), and big data ($232 billion 2011 – 2016) are increasing the need for data mining tools that process a continuous stream of events[1]. The News is filled with stories of event processing, from National Security Agency (NSA) to Facebook.

Not all event processing makes the News. Herein, I describe techniques developed to process event streams in real-time. Figure 1 illustrates the kind of analysis my research provides. Imagine a stream of events, of any kind. A data mined model, *m*, can be created for the events. As new events arrive, they can be compared with the mined model, *m*. If the new events fit within the model, then I consider them to be typical behavior; otherwise, I consider them to be interesting in some way. Much of the work described herein is based on this simple concept. To make this simple concept work in practice, my techniques address the problems of generating and comparing models from an event stream. To demonstrate the effectiveness of this framework, they have been applied to monitoring patient data in support of post-clinical care and monitoring financial transaction data for fraud detection.

[1] Market estimates from Gartner Research Inc.

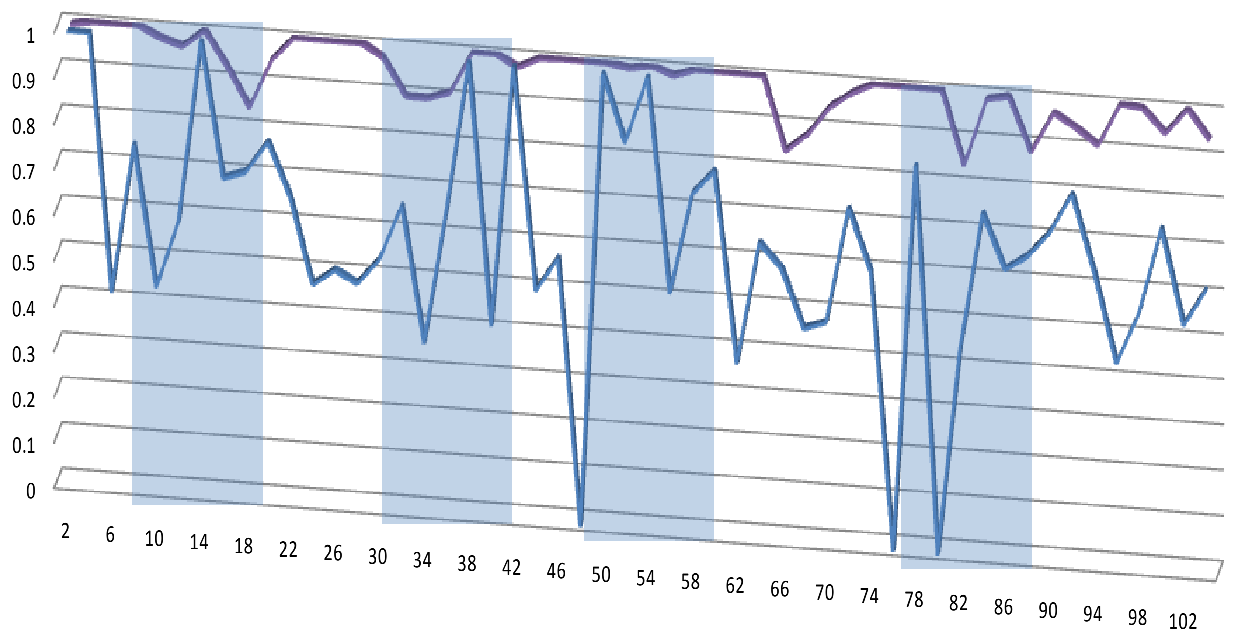


Figure 1 *Illustration stream-mined model; potential sequences containing "important events", or transitions, are shaded*

1.1. Behavioral Event-Analysis Characteristics

Data stream mining techniques can be used to monitor patient goals as part of a plan to aid cognitive rehabilitation. Workflow events, program execution events, and network traffic events can also be mined. I situate these differences within two dimensions of behavioral event analysis, as well as a four-layer framework for interpreting events as planned behavior.

Figure 2 illustrates two dimensions of data mining. The data source dimension characterizes the kinds of data processed for the discovery of patterns. The discovered patterns dimension characterizes the kind of patterns discovered by processing the data source.

As the Figure 2 indicates, data sources can be relatively focused. As an example, consider the logged data generated by tracing a program's method calls. The overarching task may be the calculation of a sales total, with each subtask applying arithmetic to sales-related objects in support of the overall calculation. Thus, the data arises from a focused task that executes on couple of entities, in single thread, in single program. Each logged event is associated with a program method that is associated with one program task. Taken as a whole, the data can be seen to rationally support the program's task.

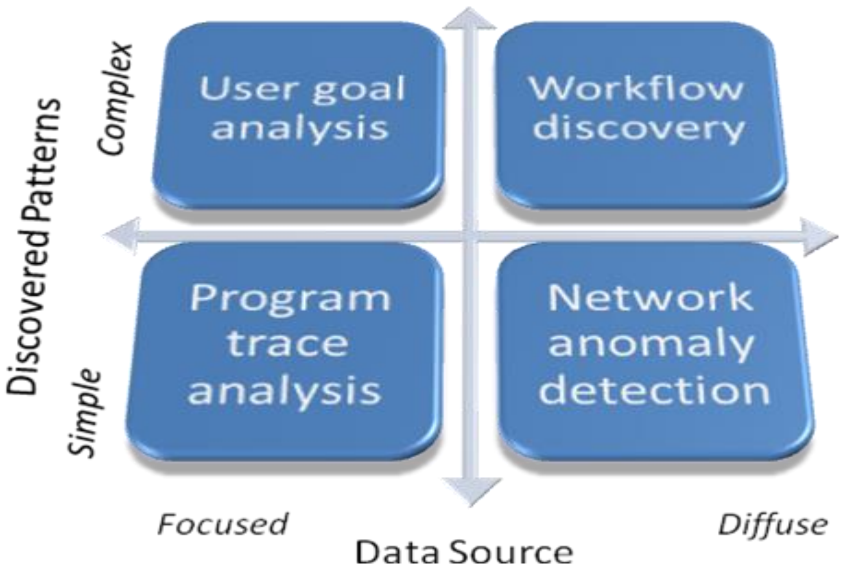


Figure 2 *Two dimensions of behavioral event-analysis*

In contrast to program trace analysis, network anomaly detection considers data from multiple programs each having multiple threads that can apply to many complex entities [1] [2]. Each program itself may be executed in support of a single user that is working on multiple overlapping tasks. Although each logged event should be associated with one program method, many events look alike; therefore, tracing from a logged event back to a program method can be problematic. This is exacerbated by the multiple programs, the multiple users, and the multiple tasks that they are performing. Thus, program trace analysis is mostly a many-to-one mapping (event to task), whereas network anomaly detection appears to be a many-to-many mapping, between events and tasks. Taken as a whole, it can be difficult to see how the data rationally supports some overarching task.

Given this mapping of low-level program events to high-level user tasks, I assume that analysts will seek to discover patterns of high-level tasks. Finding high-level task patterns is more difficult in the network analysis context because of the uncertainty in mapping low-level events to high-level tasks. For example, three login requests and three login denials may be one attacker for a single account or three forgetful users. Unless the data is properly encoded and interpreted, finding the task patterns in the event data is problematic. If the data includes complete login contexts (ID, password, client IP, time, etc.) and the data miner makes use of it, then the appropriate interpretation may be obtained. Networks, however, are awash in such data. Thus, the desired data is hidden among a relatively diffuse dataset. In contrast, program trace analysis data is more limited in scope.

Figure 23 presents a four-layer model of event interpretation. As shown in the figure, events can be traced back to tasks that achieve goals. High-level goals may be decomposed into lower-level goals, forming a goal tree in which each leaf-goal traces to a task. A set of interrelated tasks can form a workflow, which can be described with an activity diagram or other process model. Figure 3 shows that the same events can be interpreted as achieving two different goals: (1) a user is logging into their computer, or (2) a hacker is breaking into a computer. Moreover, in both cases, more than one user may be associated with the events.

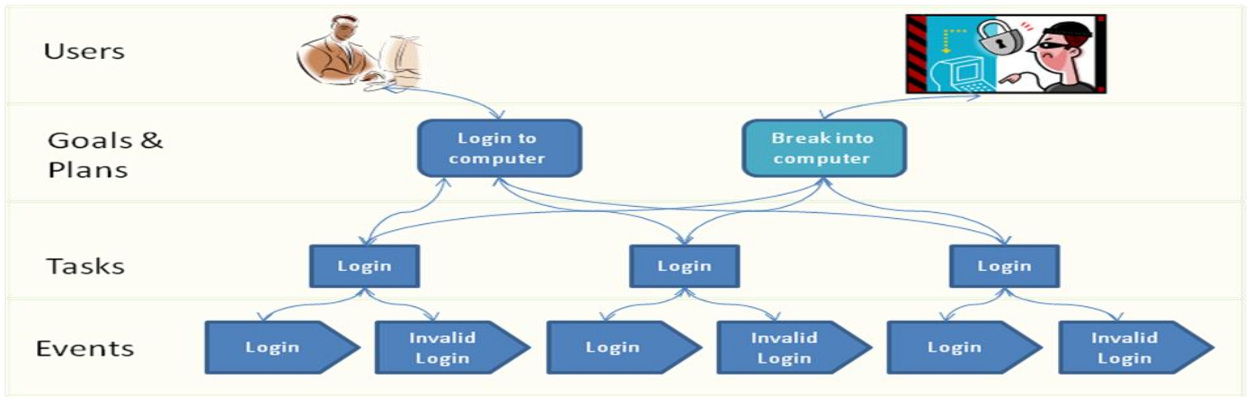


Figure 3 *A four-layer framework for interpreting events as planned behavior*

By looking back at Figure 2 we can see the significance of pattern complexity. Patterns can be simple; for example, exists event X. Patterns can be complex, referring to data and temporal relationships among events and their relationships to tasks and goals; for example, after exists event X, eventually event Y, where X.a = Y.b and user(X) = user(Y). As an illustration, consider the complex workflows of business-to-business e-commerce systems. The logged network events contain the complex data and temporal relationships of long-term of business workflows.

The user-goal analysis quadrant from Figure 2 contains much of the complex data and temporal relationships common to business workflows, but the data originates from one user and usually one program applied to the achievement of a few goals. Thus, user-task analysis has high pattern-complexity and a focused data source.

1.2. Discovery of Behavior Patterns in Event Data

The preceding two dimensions of behavioral event analysis are addressed by a variety of tools, from data mining to classic quantitative methods. We are most concerned with automated analysis—in particular, the discovery of behavioral patterns of some clinical patients.

Software Engineering has addressed the southwest and northeast quadrants of Figure 2. Tools have been developed to find predefined patterns in program event streams. Perhaps because of the domain, the patterns specify relatively simple data relationships and mostly lack temporal relationships. Some initial progress has been made into that discovery of workflows from program events [3] [4] [5]. The discovered patterns, however, have limited workflow complexity and no temporal relationships.

Data Mining has addressed the southeast quadrant of Figure 2. Mostly, the patterns are predefined because of the overwhelming amount of data and the critical nature of the anomalies; it is very important that all anomalies be identified. Moreover, the diffuse and abundant data lacks an overarching goal, making classification into a single model difficult.

Data Mining has also addressed concept drift [6]. When the defining features of a concept slowly change over time, software can automatically adjust the model to fit the new concept, which represents the "new typical". User-goal analysis is relatively less supported. The data is focused and can be rationalized as planned behavior. The data complexity and temporal dependencies within the data and behavioral model complicate analysis. As an illustration, consider the event stream generated by a user learning a word processor. Editing a document includes many planned behaviors. Tasks have dependencies, such as selecting a region before applying a command such as deleting. A user may routinize a common set of tasks and then switch to another set of tasks as she learns new commands. Thus, the user switches from one set of patterns to another set of patterns. This is not a classic concept drift, but rather a transition from a novice to an intermediate user. Moreover, users may forget and then return to the prior routinized plans. The transition from novice to intermediate back to novice and then onto expert is itself a pattern. Analysis of these meta-behavioral patterns depends on a history of behavioral models. Thus, the complexity of the user’s planned behaviors and her changes over time distinguished user-goal analysis from the simpler program event analysis. Of course, discovering workflow patterns and in their changes over time can have all of these complexities and more.

1.3. Importance of the Proposed Research

Data mining streams of events is important. It can help discover fraud and monitor patients, each of which cost our economy millions of dollars each year. For example, credit card fraud costs $5.5 billion annually. By applying appropriate techniques, event stream monitoring can discover fraud in digital transactions as it occurs. Data stream monitoring can also notify caregivers when patients change their behavior in unexpected ways.

Approximately 54 million individuals have disabilities, making up 19 percent of the U.S. population. More than one million adults in the U.S. are diagnosed each year with cognitive impairments (CI) due to neurological disease or trauma (e.g., traumatic brain injury, stroke, tumor, epilepsy, infectious disease). Currently, there are between 13.3 to 16.1 million Americans living with chronic brain disorders and associated CI [7]. Incidence rates are expected to rise due to the development of dementias associated with an aging population and increased survival rates associated with improved medical management [8]. In addition, approximately 4 million Americans have developmental disabilities that impact cognitive functioning [9] [10]. Cognitive impairments prevent this large and growing segment of our society from fully integrating into society; they are unable to participate in mainstream computer-based activities [11] [12]. There have been efforts to help this type of patients. Many researchers focused on helping them using Assistive technology (AT).

Assistive technology (AT) should help. However, Studies have found that AT systems are abandoned by CI users at shockingly high rates [13] [14] [15] [16]. One of the major causes of abandonment is an eventual misalignment with: (1) user goals and abilities and (2) the functionality delivered by the system. Data mining techniques been utilized by some researchers for mitigating some of these problems but there not been a reasonable amount of success in this regard.

1.3.1. Think and Link (TAL) Research

The five-year Department of Education project called Think and Link (TAL) developed and evaluated the effects of email interfaces that are usable by people with impairments in memory and learning as a result of brain injuries. The project produced prototypes of (1) an assessment process called Comprehensive Overview of Requisite Email Skills (CORE) and (2) the Think and Link (TAL) email interface. TAL, its commercial successor called CogLink, and its supporting analytic technologies (including that described herein) have informational web sites2. TAL derived new personalized design principles from the Think and Link project, including the following, which are critical to success.

An individual assessment process, e.g., CORE, is needed to glean the specialized goals, skills, and obstacles of each CI user (hereafter, simply user). This is not to say that each user has a unique context. Instead, each user has a personal context, i.e., there is no one-size-fits-all design for the population [17].

- User goals can be divided into those that can be achieved from the outset, and those that must be deferred until more favorable conditions exist. Deferred goals provide powerful

[2] www.think-and-link.org, www.coglink.com, eeat.cis.gsu.edu.

guidance for both follow-on training and adaptation.

- Training must be integrated into the (holistic) system design. Training provides new skills to a user. As training progresses, the user may become more capable. As the system is adapted to reflect the new user skills, new training issues arise with the added system complexity. There is a cycle of learning, system adaptation, and training.

Projects following the TAL principles include the following steps:

- A multi-disciplinary team, including cognitive rehabilitation specialists, assess a patient's needs and specifies a rehabilitation plan that includes use of AT. Clinical goals such as increased brain function are refined into subgoals that are desired by the user (i.e., user goals) and supported by the AT, such as use email to communicate with friends. (Rehabilitation specialists believe that social emailing increases brain function.)
- The patient becomes a user of the AT software system. The use of the AT is monitored and results are continually analyzed by the team. If the user attains a user goal (e.g., email a friend), then the user is encouraged to attain new goals. A user may also fail to achieve previously attained goals (i.e., forget) in which case they are encouraged to re-attain them.
- The AT system is changed (adapted) under the direction of the team, as a means to assist the user in attaining goals supported within the AT.

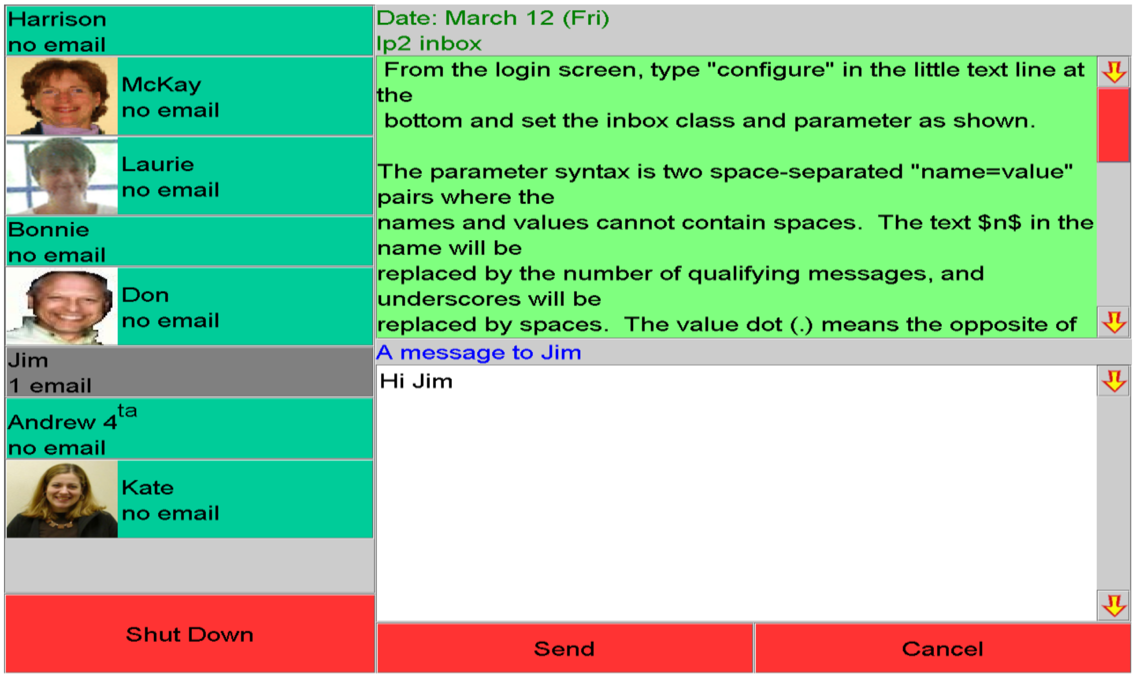
Thus, while a patient is a user of the AT, he or she seeks to satisfy his or her own personal goals (e.g., use email to communicate with friends), which in turn satisfy clinical rehabilitation goals (e.g., increased brain function). Monitoring AT usage is critical to ensure that the patient will not reject the AT (i.e., quit emailing) and will continue to attain new personal goals, and thus achieve the clinical rehabilitation goals.

1.3.2. A Cognitive Rehabilitation Scenario

Assume that Jill is interested in learning to use email. Jill acquired a brain injury in an auto accident, and has impairments in both memory and executive functions rendering it difficult to learn new skills. Jill has no memory of using a computer in the past, although her closet contains several computers, which were given by friends and family. Jill is unable to use her computers. She decides to work with a TAL staff member, Andrew, to explore the use of email. Andrew uses the CORE process to obtain two important items [18]: (1) Jill’s personal goals for using email, and (2) Jill’s existing skills for using email independently. (Other information is obtained but omitted here for brevity.) Using this information, Andrew produces a user profile containing (1) a specification of an initial system to deliver to Jill, and (2) a training plan broken into a set of lessons. Notice that Jill’s personal goals play an important role here (and will continue to play a role in the future): they filter from all possible skills the subset that is necessary to meet Jill’s needs. In essence, this is goal-directed training. Similarly, the system that is delivered is one that fits with both Jill’s current skills and her personal goals—some of which are deferred. It’s important to note that Jill has high aspirations for her use of email. She would eventually like to contribute articles to the online newsletter, published by a local group that advocates for the disabled. This goal, however, is not realizable given her current skills. Thus, it becomes a deferred goal that will be monitored.

Next, the email system is delivered to Jill. A family care-provider, Jill’s daughter-in-law Ann, assists in the training task. Soon, Jill is busy using the email system to reconnect with family and friends that have dropped out of touch. Both Jill's daily usage, and her training sessions with Ann, produces raw data. This data includes that which is generated from the email system itself, along with Ann's input on training progress. Data is also collected periodically from Jill and her email buddies through online questionnaires.

To illustrate the TAL context, consider Figure 4, which is a snapshot of a TAL email interface. This configuration is minimal in terms of functions and prompts, as required by users. As can be seen, there are eight email buddies. The email system is closed—only accessible to the user's buddies. The interface must adapt with the user. For example, when the user becomes successful—even bored—with emailing, then a new buddy may be added to the list. To avoid device abandonment, TAL automatically and continuously adapts the TAL email system to individual users—each user has their own continuously personalized email interface. To inform the adaptation process, an automated method for analyzing CI behavior with respect to their goals is a prerequisite.



From the login screen, type "configure" in the little text line at the
bottom and set the inbox class and parameter as shown.

The parameter syntax is two space-separated "name=value" pairs where the
names and values cannot contain spaces. The text n in the name will be
replaced by the number of qualifying messages, and underscores will be
replaced by spaces. The value dot (.) means the opposite of

Hi Jim

Figure 4 *One configuration of the TAL email interface*

As a user operates TAL, events are logged and then analyzed in support of decision-making about deferred goals. Working backwards, Jill has goals that are not satisfied currently. Each of these goals has preconditions, in terms of skills, that enable them. Each skill, in turn, has measured activities that signal the learning and retention of the skill. These measures can be evaluated from the collected data. Raw data is monitored. This data is evaluated for evidence of existing, and eventually, new skills, shown as an arrow back to Jill's skill set. Eventually, a match is made between a goal deferred and the skills necessary to achieve it. At this point, email system adaptation becomes the means to enabling the goal. With Jill's deferred goal of contributing to the online advocacy newsletter, the following skills are among those required: {be able to initiate email, avoid inappropriate language, keep email within a given length, use proper spelling and grammar}. Evidence for these skills can be measured. Suppose that all but the last skill have been demonstrated. At this point, two options are possible: (1) continue to train Jill in spelling and grammar, or (2) provide automated support through her email system. A spelling checker and a grammar checker are two of the functions that can be added to Jill's system. A decision is made to adapt Jill's system to include a spell checker, deferring the decision on the grammar checker until a later point. Two explicit products come from the adaptation: (1) a newly designed system that includes spell checking, and (2) additional training lessons to accommodate the new design. The implicit outcome is forward progress toward the needed skill, and hence, progress toward a deferred goal.

It is important to note here that we take a holistic system view of the problem: adaptations can occur at the software-architecture level, but also at the social-human level. The system configuration is much more than simply the software pieces. If we were to continue this scenario, we would form a cycle: deferred goals are achieved, new goals may arise, and other goals may be dropped. Although the examples are optimistically forward driven, it may be that certain skills are lost over time, making system retraction a real possibility, thus adapting the system to a less complex version rather than a more complex version.

1.3.3. Prominent Issues for Personalized Cognitive Rehabilitation

It is important to highlight two issues from Jill's story.

- Composite system design for each individual is necessary [19] [20]. The software component of a system plays an important role. However, the human context is equally important. Looking at Jill's example, her careprovider and her email buddies are components in the system design space. These components can be configured (e.g., add/remove buddies, add/remove careproviders). We can also attempt to influence their behavior in the system (e.g., through prompting).

- Real-time, monitored, data analysis is critical. Users can and will abandon poorly designed AT systems at any point along a timeline. It behooves us to monitor their progress, noticing obstacles and achievements. Logging and reviewing data is necessary, but not sufficient. Facing shorten decision cycles and increasing data volume, business activity monitoring (BAM) or real-time business intelligence (RT-BI) supports business management in guiding the real-time organization [21] [22] [23] [24]. Similarly, caretakers need support in analyzing voluminous patient data to guide their personalized treatment. They need a kind of micro-usability monitoring that tracks individual user behaviors on the scale of minutes to months. This is particularly challenging, as we must hypothesize monitored properties or discover behavioral patterns that serve as proxies for our understanding of a patient's behavior. Recognizing a significant change is most important, from clinical rehabilitation to trauma treatment. Understanding the nature of the change is next in importance. Finally, intervening appropriately is often necessary.

These two research issues are central to the TAL project, and cognitive rehabilitation more generally [25]. Our role in the TAL project has been to provide monitored real-time analytics. Figure 5 presents the kind of analysis being automated. Consider the line graphs as a representation of consistent behavior. If y=1 for all x, then the user does exactly the same behavior for each time-point x. (These charts are explained in section 3) The sharp dips in the graph are the significant points of interest (e.g., x=10, x=50). They suggest that the user substantially changed his or her behavior. Of course, one difficulty is to distinguish minor daily variations from significant transitions in learned behavior. The automated analysis reveals these potential goal transitions and the automated diagnostic technique presents the behavioral differences. For example, (1) there is a potential transition around week 10, and (2) the change in behavior is a 6 percent reduction in sending email around 10 P.M. I describe the data mining techniques beginning in section 3, after a brief review of related research.

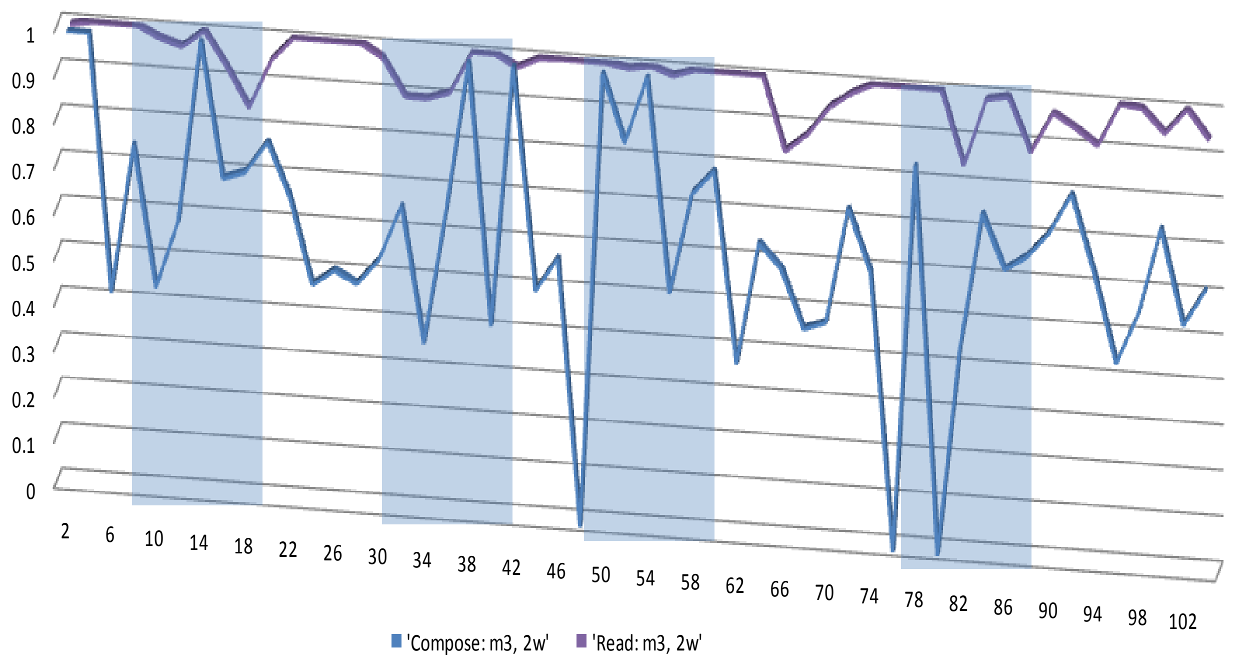


Figure 5 *Quality of stream-mined models for read and compose email, over 2 years of data using 2-week windows; potential goal transitions are shaded*

1.4. Progress of TAL Data Analysis Techniques

Analyzing the use of the AT is critical, according to the TAL principles. Over the years, monitoring has progressed for TAL and its subsequent projects:

- Emailing (AT) usage is logged. Early on, these logs were manually reviewed.
- Custom coding provides analysis of the logs, based on what the team requests. For example, a graph of daily emailing counts is presented.
- Logical properties about the events are determined by a property monitoring system. A software toolkit simplifies the acquisition and logical analysis of events [26] [27]. For example, one can monitor the temporal property: After receiving an email, a user shall

read and reply to the sender, within k days. (The real-time, temporal, context-sensitive nature of this proposition is not normally addressed by commercial monitoring systems.)

- Potential transitions in user behavior are identified by data mining the event stream using decision trees, sequence –mining algorithm and hidden Markov models [28].
- Behavioral differences at transition points are analyzed through model differencing—the results are used to assess the need to adapt the system to meet changes in user behavior [29].
- Potential transitions in user behavior are identified as learning or non-learning using a technique utilizing hidden Markov models.

The last three techniques provide new ways to address real-time data-analysis problems. This research focuses on the last three technologies, which rely on data mining techniques.

1.5. Research Objectives

Over the past 10 years, a multi-disciplinary group of cognitive psychologists, computer scientists, and clinical workers have been successfully delivering AT to CI patients [30] [31] [18] [32] [33] [34] [35] [25] [8] [17] [36] [37] [38] [39]. As researchers, our role within this group has been to provide real-time data analytics. By combining and extending data mining and stream mining techniques, this research has provided a series of patient-monitoring software systems with ever-increasing functionality.

In summary, this research answers three important research questions (RQs) of real-time data analytics:

- RQ1: Can software automatically recognize changes in patient behavior, where a patient’s behavior is defined by the stream of events generated from his or her use of software? This research affirms that recognition can be automated using several different techniques such as decision-tree, sequence mining and hidden Markov models.

- RQ2: Can software automatically diagnose causes of patient's behavioral change, in terms of the changes in their event stream? This research affirms that diagnosis can be automated using differential analysis of data mined models.
- RQ3: Can software automatically determine if a change in a patient's behavior is long-lasting, as opposed to transient? This research affirms that detection of patient learning can be automated using differential analysis of data mined models.

The focus on automated, real-time data analysis is by necessity. Automated, accurate, real-time data analysis is critical to meeting the growing demands on health-care providers (as well as other domains, such business activity monitoring) [23] [24] [40]. Real-time analytical techniques, like that presented herein, enables fewer healthcare providers to meet the needs of more patients. Consequently, improvements in this area may affect millions of individuals.

These technologies have implications for the analysis of planned behavior [5] [19]. Consider, for example, business processing in support of order fulfillment. The sequence of events generated with each business process represents the planned behavior of the organization in fulfillment of the goal to process orders. Similarly, a computer hacker generates a sequence of events, such as improper Logins, in an effort to fulfill the goal of a system break-in. In both cases, an analytic technique that discoverers transitions from routine behavior will be helpful to identify potential problems. Although the techniques have been applied to patient monitoring, they may also apply to business processes, network monitoring, fraud detection and many other areas with similar nature. The techniques were developed in the patient monitoring domain. To demonstrate their generalizability, they were also applied in fraud detection within financial transactions. Section 4 illustrates the procedure and results for fraud detection.

2. LITERATURE REVIEW AND RELATED RESEARCH

This section provides a summary of my literature review on relevant event stream analytics from academic, practitioner, and Internet sources. I searched various literature databases such as ACM, IEEE, and Google Scholar. The keywords used a combination of key words including: data stream-mining, high speed mining, limitation, modeling, restriction, issues, challenges, *etc*. I gathered about 100 relevant articles. After briefly reviewing the articles, I filtered them to the 30 most relevant.

2.1. Characteristic of Various Stream Mining Techniques

After reviewing these articles, I categorized the main stream mining issues systems might encounter. The following techniques have been identified based on a review of the literature:

2.1.1. Solution Approach: Data-based vs. task-based techniques

Gaber distinguishes computational and statistical techniques into two broad areas: data-

based and task-based [41]. The former includes sampling, load shedding and sketching and synopsis techniques, which operate by building summaries on the entire dataset or some subset of the data. The latter refers to techniques developed in response to computational time and space issues, and includes approximation algorithms, sliding windows and algorithm output granularity.

2.1.2. Classification Techniques

Gaber moves from solution approaches to classification methods and discusses seven specific stream classification techniques based on a review of the recent literature. The techniques are summarized and contrasted in three areas: ability to deal with concept drift, high speed streams and unbounded memory requirements.

2.1.3. Online, Offline and hybrid techniques

Application requirements determine whether mining is performed online, offline or in stages, for example in the OLAP and clustering scenarios. The specific techniques used are further determined by the amount of processing and storage available.

2.1.4. Interactivity of mining

In some cases, the domain may require that an analyst drive the mining process, for example to drill down in an OLAP scenario [42]. The approach taken here is similar to the clustering scenario mentioned above. The mining may be split into online and offline stages, where the first stage mines sufficient information to build a data cube, for further processing by an analyst in the offline stage. The latter stage is affected the level of historical data required for analysis. Some techniques include the use of tilted windows or pyramidal time-frames to reduce the size of data required.

2.1.5. Data & Window models

Windowing techniques are instrumental in building snapshots of data streams to mine the changes over time. The choice of windows is influenced by the available storage for online or offline processing. The data model influences the choice of windows in three ways. Intervals may be fixed or one or both endpoints may move. This leads to fixed windows, landmark windows or sliding windows. Windows may be time-based (physical), based on a time interval, or count-based (logical), based on a number of tuples.

2.1.6. Required level of accuracy/precision

This is influenced by both device characteristics and domain requirements. There may be significant bounds on memory or processing. In addition, application requirements may specify requirements based on timing or precision of results. The algorithms used are then constrained by these requirements. In particular, approximation algorithms may be required, which provide answers within specified error bounds.

2.1.7. Evolution of mining results

This is distinguished by whether the application requirements are satisfied by the mining results at any given time, or the nature of evolution of those results. For example, in a stream clustering application, rather than determining the clusters at any given time, we may be interested in how the clusters evolve from one time frame to the next. In this case, we may require two stages of processing: online and offline. The online stage performs micro-clustering to determine sufficient statistical information for the offline macro-clustering stage [3]. The device characteristics, especially available storage, influence the choice of such techniques. Another example is velocity density estimation used to analyze and predict trends, where temporal and spatial velocity profiles are built over the stream and used to predict three types of data evolution: dissolution, coagulation and shift [3].

2.1.8. Granularity of analysis

This is determined by the application requirements and specifies the levels of granularity of mining results. For example, in the OLAP scenario [42], two levels may be defined: the observation layer, where the analyst operates at the offline stage, and a minimal layer of interest, where sufficient results are mined at the online stage providing the lower bounds for drilling down.

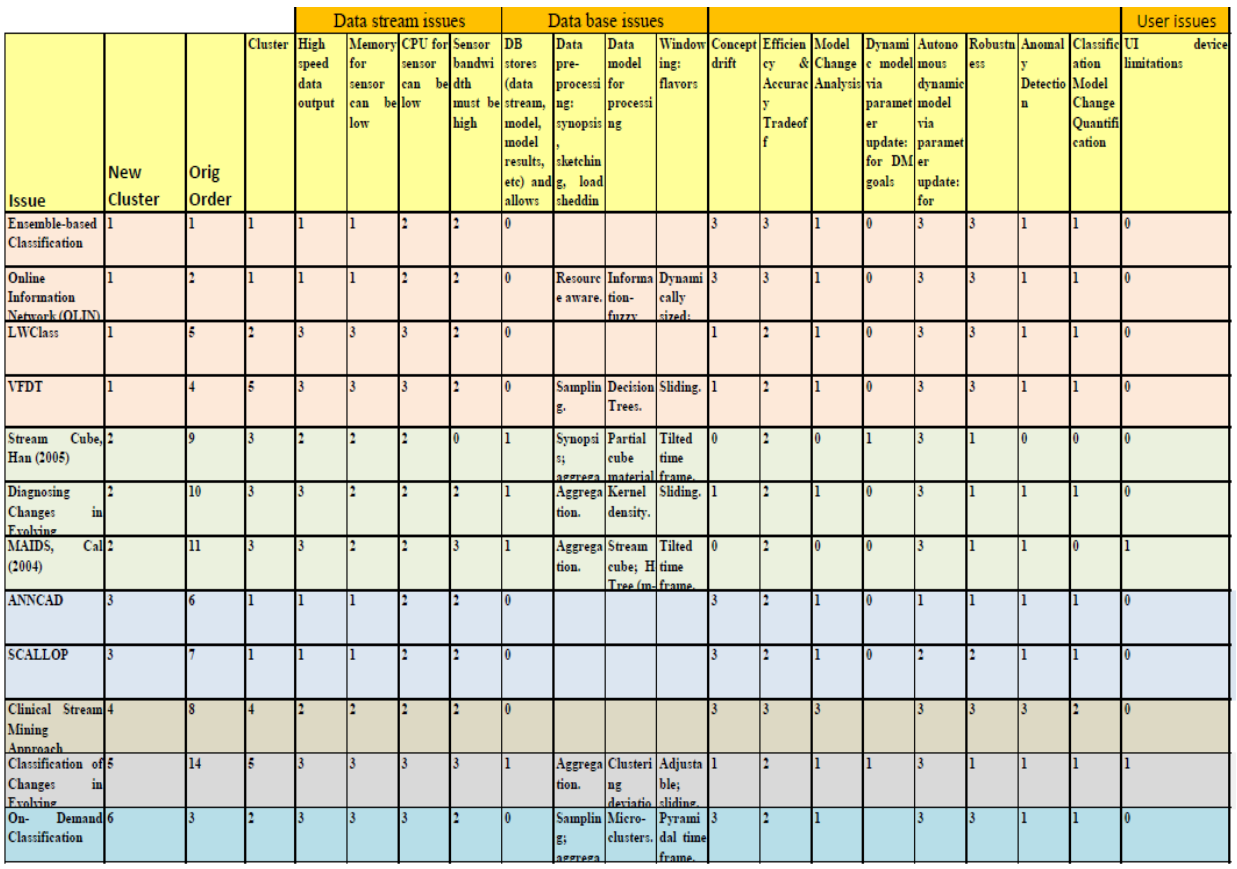
2.1.9. Stream queries

Queries may be predefine or ad-hoc. Predefined queries are formulated in advance of the arrival of data, whereas ad-hoc queries are formulated online after the data begins to arrive. The latter is the concern of stream management systems, and affects query optimization. In particular, ad-hoc queries are limited by how much historical data may be referenced efficiently. The choice of querying techniques is also influenced by how data is maintained in memory. Querying may involve updating the maintained data with the arrival of new data, and computing the result based on the new data. The time taken for each operation determines the choice of algorithms. Fast update and slow compute times lead to a batch processing technique, where time is traded off for accuracy, useful for bursty streams. Slow update and fast compute times lead to sampling algorithms, where an approximate answer must be used when data arrives faster than it can be processed. Fast update and compute times suggest synopsis techniques, where small summaries of data can be maintained for efficient processing.

2.2. Clustering Various Stream Mining Methodologies

Several mining methodologies found in selected literatures and their processing techniques and characteristics mentioned in previous section being investigated. The aim of this was to see how different these mining methodologies are from each other and if the clinical stream mining approach can be categorized as a new methodology in this research domain. Based on this information, I prepared a table that the columns are stream mining classification algorithm and the rows are the issues we deal with in data stream-mining. The values in each row and column indicate how supportive the algorithm is toward the mentioned issue.

Table 1 *Characteristic of various stream mining methodologies based on stream mining techniques*



				Data stream issues				Data base issues												User issues
Issue	New Cluster	Orig Order	Cluster	High speed data output	Memory for sensor can be low	CPU for sensor can be low	Sensor bandwidth must be high	DB stores (data stream, model, model results, etc) and allows	Data pre-processing: synopsis, sketching, load sheddin	Data model for processing	Windowing: flavors	Concept drift	Efficiency & Accuracy Tradeoff	Model Change Analysis	Dynamic model via parameter update: for DM goals	Autonomous dynamic model via parameter update: for	Robustness	Anomaly Detection	Classification Model Change Quantification	UI device limitations
Ensemble-based Classification	1	1	1	1	1	2	2	0				3	3	1	0	3	3	1	1	0
Online Information Network (OLIN)	1	2	1	1	1	2	2	0	Resource aware.	Information-fuzzy	Dynamically sized:	3	3	1	0	3	3	1	1	0
LWClass	1	5	2	3	3	3	2	0				1	2	1	0	3	3	1	1	0
VFDT	1	4	5	3	3	3	2	0	Sampling.	Decision Trees.	Sliding.	1	2	1	0	3	3	1	1	0
Stream Cube, Han (2005)	2	9	3	2	2	2	0	1	Synopsis; aggrega	Partial cube material	Tilted time frame.	0	2	0	1	3	1	0	0	0
Diagnosing Changes in Evolving	2	10	3	3	2	2	2	1	Aggregation.	Kernel density.	Sliding.	1	2	1	0	3	1	1	1	0
MAIDS, Cal (2004)	2	11	3	3	2	2	3	1	Aggregation.	Stream cube; H Tree (m-	Tilted time frame.	0	2	0	0	3	1	1	0	1
ANNCAD	3	6	1	1	1	2	2	0				3	2	1	0	1	1	1	1	0
SCALLOP	3	7	1	1	1	2	2	0				3	2	1	0	2	2	1	1	0
Clinical Stream Mining Approach	4	8	4	2	2	2	2	0				3	3	3		3	3	3	2	0
Classification of Changes in Evolving	5	14	5	3	3	3	3	1	Aggregation.	Clustering deviatio	Adjustable; sliding.	1	2	1	1	3	1	1	1	1
On-Demand Classification	6	3	2	3	3	3	2	0	Sampling; aggrega	Micro-clusters.	Pyramidal time frame.	3	2	1		3	3	1	1	0

As it can be seen in the Table 1 each methodology for any date stream or database issue marked as 0 (Not supportive) – 3 (Fully supportive). Using the information from this table, I used clustering to bundle the methodologies into similar groups. I used k-means clustering algorithm and the results ended clustering the methodologies into 6 clusters.

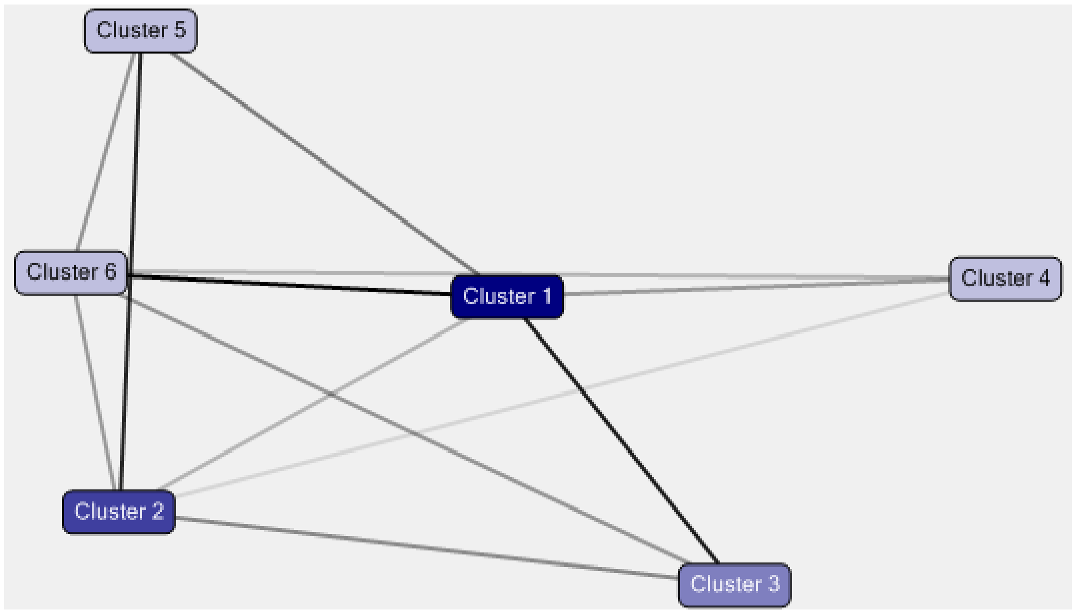


Figure 6 *Six Cluster of various stream mining methodologies*

Table 2 *Cluster of various stream mining methodologies based on stream mining techniques*

Methodology	**Cluster**
Ensemble-based Classification	1
Online Information Network (OLIN)	2
Ensemble-based Classification	1
Online Information Network (OLIN)	1
LWClass	1

VFDT	1
Stream Cube, Han (2005)	2
Diagnosing Changes in Evolving Streams, Aggarwal (2003)	2
MAIDS, Cal (2004)	2
ANNCAD	3
SCALLOP	3
Clinical Stream Mining Approach	4
Classification of Changes in Evolving Streams, Gaber	5
On- Demand Classification	6

Cluster 1

This cluster includes Ensemble-based Classification, Online Information Network, LWClass and VFDT.

In the "Ensemble-based Classification", they used the idea of not just classifying on whole data set or fixed size sliding window but using snapshots on some part of data (small history of models) and then use all of them to classify the label field and pick the prediction which is in majority. It also describes a technique that how to update the history of models by passing through data.

Regarding "Online Information Network", most classification methods are based on the assumption that the data conforms to a stationary distribution. However, the real-world data is usually collected over certain periods of time, ranging from seconds to years, and ignoring possible changes in the underlying concept, also known as concept drift, may degrade the predictive performance of a classification model. Moreover, the computation time, the amount of required memory, and the model complexity may grow indefinitely with the continuous arrival of new training instances. OLIN, an online classification system, dynamically adjusts the size of the training window and the number of new examples between model re-constructions to the current rate of concept drift.

LWClass stands for Lightweight classification. LWClass starts with determining the number of instances that could be resident in memory according to the available space. Once a classified data record arrives, the algorithm is going to search for the nearest instance already stored in the main memory. This is done using a pre-specified distance threshold. This threshold represents the similarity measure acceptable by the algorithm to consider two or more data records as an entry into a matrix. This matrix is a summarized version of the original data set. If the algorithm finds a nearest neighbor, it checks the class label. If the class label is the same, it increases the weight for this instance by one, otherwise it decrements the weight by one. If the weight is decremented down to zero, this entry will be released from the memory conserving the limited memory on streaming applications.

VFDT (Very fast decision tree), uses dynamic size-window and an algorithm for faster classification. Because this methodology uses dynamic window it includes the effect of concept-drift more into creating the models. Using dynamic window will cause the window size decreases tremendously and increase in the number of windows. This causes slower model creation and classification. The methodology introduces a faster algorithm for solving the issue of the slow speed. Again, what the methodology does not bring into consideration is using batches with multiple sizes and as mentioned before not using this has its own short fall.

All these approaches are robust and almost have a good support for efficiency/accuracy trade-off. They all not have good support for anomaly detection, model change analysis and classification model change quantification. The only processing issue that makes Ensemble-based Classification, Online Information Network different from LWClass and VFDT is concept-drift. The first two approaches have good support for concept-drift and the second two do not.

Cluster 2

The systems in cluster 2 are primarily concerned with change detection in data streams. All three systems involve some type of user driven approach for classifying change. For example, Stream Cube requires an analysis layer where the user may drill down to obtain further data. At least two of the systems use some type of tilted window for summarizing and detecting changes. In general, none of the systems have specific memory or CPU requirements, but all are able to cope with high stream data.

Cluster 3

In “SCALLOP”, the algorithm starts by reading records which are labeled by the user. Then a number of rules get deducted for each class from the records. Subsequently, the key issue is to maintain the rule set after arrival of each new record. After reading a series of records, the refining process is performed. The rules in the same class and within the acceptable distance measure are merged. At the same, care is taken to ensure that these rules do not intersect with rules associated with other class labels. In “ANNCAD” (Adaptive Nearest Neighbor Classification for Data-streams), it uses Haar Wavlets transformation for multi-resolution data representation. A grid-based representation at each level is used. These two approaches have very good support for concept drift and good support for accuracy/efficiency trade-off. They don’t have good support for other issues especially for anomaly detection, model change analysis and classification model change quantification.

Cluster 4

The Clinical Stream Mining Approach is the technique I am going to use in my research for monitoring and analyzing cognitively impaired patient behavioral changes. I used sliding window approach with different fixed size window for each run over the whole patient data. Each window model was created using three different decision tree algorithms (Ensemble model) to provide us with best accurate model. Each fixed window would provide us an accuracy/precision graph. Analyzing the overlapped graph could provide us a good indication of change of patient behavior on some highlighted points in the whole time period of the patient activity [29].

The “Clinical Stream mining Approach” can deal with some issues that other techniques are not or little able to deal with. These issues are anomaly detection, model change analysis and classification model change quantification. The last issue can be handled by this approach if I use some OLAP and other analyzing techniques. That is why it is mentioned in the chart as being able to support but not fully supported.

Cluster 5

Cluster 5 has one system and is also concerned with change detection. It can specifically cope with concept drift. One distinguishing feature is its use of clustering to classify changes and examining differences in clusters over time to classify their evolution. So this system is concerned not only with change detection but also with the evolution of change.

Cluster 6

Cluster 6 also uses a clustering approach for detecting and classifying change and is similar to cluster 5. However, it is different in its use of tilted windows to summarize and classify changes. Like the Stream Cube system in cluster 2, it uses a two-stage approach, building a minimal layer followed by a analysis layer which may be manipulated via user interactions.

Most data stream mining application being utilized in industry or in different research areas try to tackle or investigate one of the issues mentioned in my survey. The clusters created from these methodologies very well reflect the fact that each cluster can be supportive of some of stream mining issues but not all. Processing issues are one of major thing that I concentrated on them. Observation on the created clusters or even looking at the processing issues we can identify that most common methodologies can being supportive of concept drift detection, efficiency and accuracy trade off and robustness. Even if some methodologies are weak in supporting some of these issues the full supportive one can be easily chosen and applied. The mentioned issues are very popular and common issues and support of them is the goal of many stream mining applications. However, there are other issues which can be enough interesting and considerable for researchers or other stream mining applications. Model change analysis, classification model change quantification hence showing meta-behavioral patterns less applicable to some domains can be the center piece of other domains. In my research, I was interested to tackle these issues in my data as these issues had key information about analyzing the patients' behavior so I came up with a methodology (Clinical Stream mining Approach) that can be good support of these issues.

Model change analysis, classification model change quantification and anomaly detection are subjects that been investigated and researched individually and are not some new issues. The Clinical Stream mining Approach I described is a supportive combination of all three mentioned issues and the findings and results can be quantified. It is important to mention that this methodology meanwhile can be good support of other issue mentioned above by other methodologies.

2.3. Summary

In this section, I surveyed different research papers and extracted the common methodologies being used for data stream mining. I also investigated the issues and limitation that these methodologies deal with. Most of these issues originate from the nature of stream mining which is very high speed in nature and huge in the volume. Using these two set of information, I created tables that reflected to what extent each methodology can deal with the stream mining issues. Eventually I clustered these methodologies and came up with six clusters.

Most data stream mining application being utilized in industry or in different research areas try to tackle or investigate one of the issues mentioned in this section. The clusters created from these methodologies very well reflect the fact that each cluster can be supportive of some of stream mining issues but not all. Processing issues are one of major thing that we concentrated on them. By observation on the created clusters or even looking at the processing issues we can identify that most common methodologies can being supportive of concept drift detection, efficiency and accuracy trade off and robustness. Even if some methodologies are weak in supporting some of these issues the full supportive one can be easily chosen and applied. The mentioned issues are very popular and common issues and support of them is the goal of many stream mining applications. However, there are other issues which can be enough interesting and considerable for researchers or other stream mining applications. Model change analysis, classification model change quantification and anomaly detection are issues that even though less applicable to some domains can be the center piece of other domains. I was interested to tackle these issues in my data as these issues had key information about analyzing the patients' behavior so I came up with a methodology (Clinical Stream mining Approach) that can be good support of these issues.

Model change analysis, classification model change quantification and anomaly detection are subjects that been investigated and researched individually and are not some new issues. The Clinical Stream mining Approach I described is a supportive combination of all three mentioned issues and the findings and results can be quantified. It is important to mention that this methodology meanwhile can be good support of other issue mentioned above by other methodologies.

3. DEVELOPMENT AND EVALUATION OF AUTOMATED DATA ANALYTIC TECHNIQUES

The main objective of this research is to develop automated, real-time data analysis that can assist caregivers to CI patients and better utilization of their assistive technology. In this section I will go through design, development and evaluation of each suggested technique.

3.1. Recognizing Potential Goal Transitions

The approach to the analysis of (CI) user behavior has been to extend classic data mining techniques. Two extensions are important. First, logged events are processed with stream-mining methods. Second, changes in mined-model qualities are used to suggest changes and user behavior. Overall, the approach automatically models user behavior based on events and recognizes significant changes in those events.

3.1.1. Goal Attainment Scales

The cognitive rehabilitation field uses a goal attainment scale to specify the individual goals and desires of a person. Each goal is broken into a set of attainment levels to provide a measure of attainment. For instance, a goal might be to be able to do shopping for meals, with this broken into degrees of attainment, e.g., can shop for all meals, can shop for special meals, etc. Using this style, individual CI goals are specified. Each user is asked to first list a goal and then five levels of attainment, ranging from not-attained to fully-attained. For example, one of Don's goals is to be socially involved through online communications. One of his goals was to learn to email with no help. He divided this goal as follows:

- Level 1 (not attained): will not be able to learn how to use email.
- Level 2: can email, but only with lots of prompting and help.
- Level 3: can email, with some prompting and help.
- Level 4: can email with no prompting and help.
- Level 5 (fully attained): can teach others how to email.

It is important to note the temporal nature of these goals, and in particular, the division of attainment levels into milestones. This is the norm in clinical fields: individuals state long-term goals and then work towards them.

Figure 7 illustrates some of Don's formalized emailing goals. Notice that the attainment levels are mapped onto subgoals in the goal tree. Don, wants to use email to engage in online social communication. This need is shown as the top-most, root goal in Figure 7 supporting emailing subgoals, such as composing and sending consistently, are shown below the root. The goal graph is typical in that the leaves specify specific monitored events, while the high-level goals aggregate the lower goals, using logical relationships, sum, average, etc. In the case of TAL, user (patient) goals are towards top of the graph, while clinical measurements are towards the bottom. Overarching clinical goals, such as increased brain function, are not shown in the graph.

Goal Transition

I am looking for a change in user behavior—more specifically, a significant change in behavior that leads to new goal attainment or the loss of goal satisfaction (i.e., forgotten behaviors). Consider a user goal set, Gi, which specifies the currently attained goals. I want to know when a user transitions to new goal set, Gj, where the difference between the user goal attainment is one goal, i.e., Gj - Gi = g.

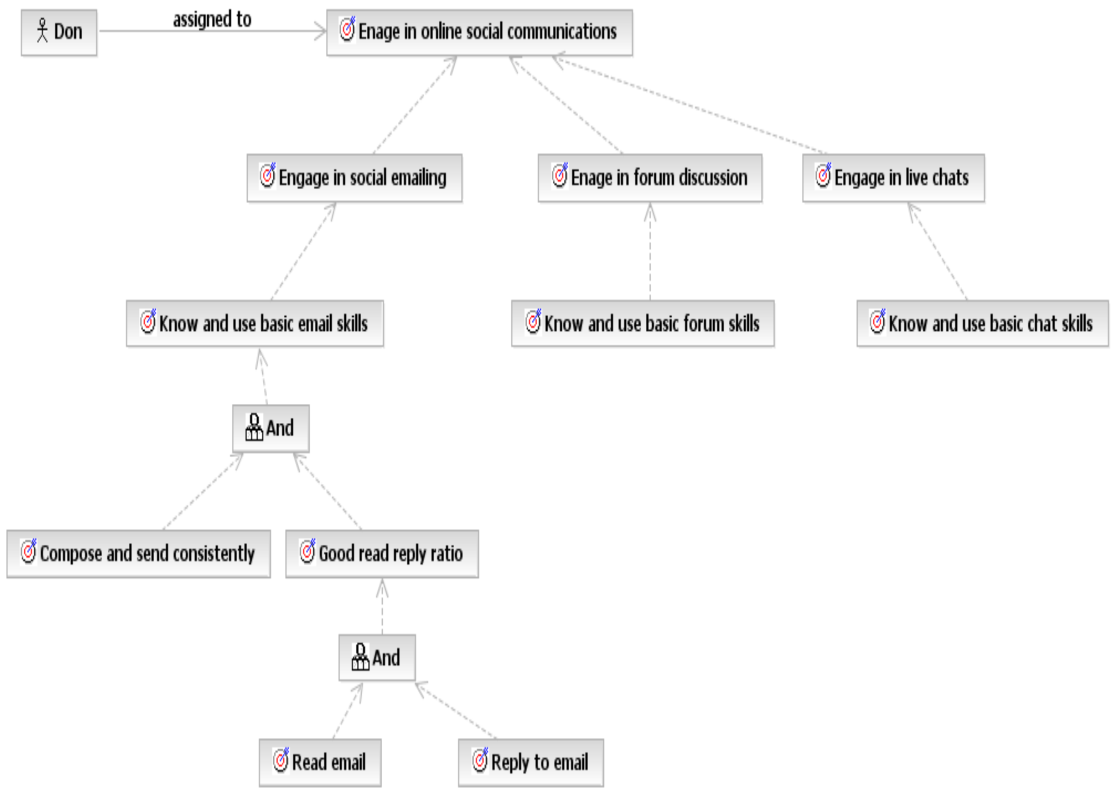

Don
assigned to
Enage in online social communications
Engage in social emailing
Enage in forum discussion
Engage in live chats
Know and use basic email skills
Know and use basic forum skills
Know and use basic chat skills
And
Compose and send consistently
Good read reply ratio
And
Read email
Reply to email

Figure 7 *Some of Don's emailing goals, as depicted by an Eclipse goal modeling plugin***.**

Around the time that goal transitions occur, a user's behavior becomes less consistent. Initially, a user engages in a stable set of behaviors. Next, as a user tries to attain a new goal, he or she will engage in new behaviors. Thus, goal transitions are associated with new behaviors. By monitoring the stream of user events, I can identify potential transitions by identifying unusual patterns of behavior against a common background. I apply a data-stream mining approach to identify unusual behavioral patterns and specific metrics to select those most likely to be associated with goal transitions.

Recognizing Behavioral Changes

Stream mining methods can be used to discover significant changes in behavior. Within the context of TAL, a user's behavior is defined as the stream of events that they initiate with the TAL emailing client. A user's behavior will often contain reoccurring patterns, which can be interpreted as planned and executed event sequences with the purpose of goal fulfillment.

Stream mining methods can be used to discover significant changes in behavior. Stream mining produces a sequence of models, m_1 .. m_n that predict the behavior observed in windows w_2 .. w_{n+1}. (Here, I assume supervised models, such as decision trees or Bayesian networks.) After a predicted window is observed, w_i, the prediction quality of its model, q(m_{i-1}, w_i) = [0,1], can be calculated. There are a number of ways in which quality can be defined. For example, the classic metrics of accuracy and precision can be used individually or in combination.

Accuracy measures how error-free the model's predictions are, according to this equation.

accuracy = (true negative cases + true positive cases) / all cases

where all cases = true negative + true positive + false negative + false positive cases

Precision measures fidelity, according to this equation.

precision = true positive cases / (true positive + false positive cases)

A model can have high accuracy but low precision, which indicates the model is not useful. As an example, consider predicting 100 patients for cancer. In the data, 95% of patients are cancer free, but the model predicts 100% cancer. The model is 100% accurate, but has 5% precision—a very poor model indeed.

Predictive quality can be automatically evaluated as windows update during stream mining. Given that model m_{i-1}, is trained over window w_i, one can evaluate the predicted values of m_{i-1} against the known data inw_i.

Consider the case where the predictive quality is nearly a constant 0.9 on a scale from 0 to 1, where 1.0 is prefect quality. This suggests that the models are good and that the behavior from one window to the next is nearly constant. Now, consider a sequence of some n predictions, with q_1 .. q_n, where each $q_i \approx 0.9$ with the exception of q_k ($1 < k < n$) which is 0.1. Again, this suggests that the models are good with the exception of m_{k-1}, which was trained on window w_{k-1} to predict window w_k. From this, I infer that something interesting happened during window w_k. That is, the events in window w_k are so different from the events of window w_{k-1} that the model trained on w_{k-1} cannot reasonably classify the new window w_k behavior. This is the technique I use to recognize significant behavioral changes.

The predictive quality change is dq/dt. Thus, $|q'|>\varepsilon$ implies behavioral changes from window w_{k-1} to window wk. I can also consider how quickly the predictive quality changes, which is q''. An analysis of typical domain values for q' and q'' can provide guidelines that distinguish typical behavioral variations from significant behavioral changes.

The particular use of stream mining and predictive quality to discover behavioral changes will be elaborated in the context of a case study in user-goal analysis (Section 3.3.4). There, I show how a negative value of q' for multiple window models suggests a behavior change.

Good Model Quality is Sufficient

A good model is sufficient for finding significant differences in a model sequence. Great or nearly perfect predictive models are wonderful, but unnecessary for identifying goal transitions. The model differences are most important. Of course, low quality models will have substantial variance from model to model in stream mining. Good quality models have less variance and the differences will be striking. The key is that the differences are more prominent than the random variance in the models. In the TAL domain, the goal has been better than 70% for both accuracy and precision.

There are many ways to improve model quality. Selecting and turning appropriate techniques is one approach. Another includes the use of multiple models simultaneously (i.e., data mining ensemble). Additionally, consideration of multiple window sizes can increase the aggregate quality. I apply these three approaches.

Selecting the appropriate window size, w_s, is important. If w_s is small, then insufficient data will be available when the mining model (e.g., decision tree) is derived. Conversely, if w_s is large, then the analyst must wait, perhaps a long time, before the model is derived—moreover, model construction itself can take a long time for large data sets. Finally, large data windows have a regression to the mean problem—short variations in behavior will be discounted in favor of more common behavior, and thus short variations may not be represented in the mined model. Thus, widow size affects precision, accuracy, and availability of the mined model.

Some data stream miners incrementally update a single model as the data becomes available. Thus, the model grows with the window size. Such an approach is good for incrementally changing the model; however, for us, we looking for transitions between models, where each model represents the learned behavior associated with a user goal. Therefore, I aim to acquire a goal model, and then notice when it changes significantly. Such significant changes may become hidden among many small, incremental model changes, but become apparent when a static model is compared with new behaviors.

My approach is to run multiple concurrent windows; for example, two-week, four-week, and eight-week. I also improve model quality by using multiple models (i.e. data mining ensemble). My approach is elaborated in the case study of section 3.3.

3.2. Diagnosing Goal Transitions

A discovered potential goal transition change begs the question, “what exactly has changed?” Diagnosing a goal transition provides an answer, which is used by clinicians to decide if a goal transition has occurred. If it has, then a system adaptation or training follows, as directed by the clinicians. Model differencing reveals the most significant behavioral changes evident between two models. In the TAL diagnosis context, it can reveal that the user “now sends emails more emails on Wednesday than in the past”. (Failing to take medications on Wednesday was an underlying cause for this real example.)

In TAL, emailing is part of cognitive rehabilitation, but only a part. Thus, there may be latent factors that influence patient-user rehabilitation. I seek to provide clinicians guidance on what are the most significant behavior changes in AT usage around a goal transition. To this end, I employ a model differencing.

Given a behavioral transition, diagnosis determines the behavioral change by determining the associated model change. Each data mined model, from the user’s data stream, characterizes the user’s behaviors. Thus, model differences characterize changes in user behavior.

A data mined model, m_i, provides a model of the data observed in window w_i. For example, a decision tree model, m_i, characterizes the data distribution observed in window w_i. When model quality falls substantially from w_{k-1} to window w_k, it means that the data varies substantially from w_{k-1} to window w_k. Thus, given a window w_k with $|q'|>\varepsilon$, three data windows are of interest: wk-1 (before change data), w_k (change data), and w_{k+1} (after change data). To diagnose persistent change, I compare the models associated with windows w_{k-1} with w_{k+1}. I consider w_k (change data) of lesser importance because it represents the transition from one stable goal state to another. Next, I illustrate decision-tree model comparisons.

3.2.1. Decision Tree Differencing

Figure 8 illustrates a portion of a decision tree that models user email events. For a data window, the model summarizes the kind of email event (Event: read, receive, delete, compose), the day of the week (weekday: 1-7), and a 2-hour period in which the event occurred (DaySegment: 0 - 11). The leaves of the tree represent the email events. A path from the root to a leaf can be considered a query (or proposition) that characterizes the leaf data. For example, the path DaySegment = 4, weekday = 1 leads to the events satisfying those two attribute values. In Figure 8, the data distribution is represented by the colored bar chart on the leaves. Thus, the bottom left leaf shows that the path DaySegment = 4, weekday = 1 has only read email events. On the other hand, other leaves have a mix of colors, indicating a mixed distribution of email events. (The node is labeled with the dominant event number—read = 1, compose = 5.)

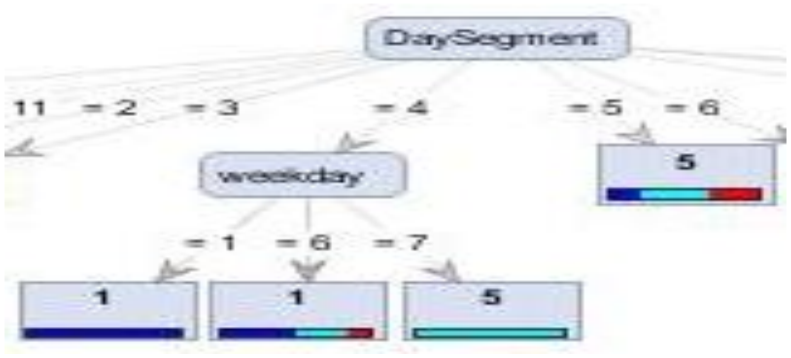


Figure 8 *A decision tree classifying compose events*

To illustrate model differencing, consider two hypothetical decision trees. The first has only read events for day segment 1, while the second has only read events for day segment 2. The difference of the two models (m_2 - m_1) reveals that read has shifted from DaySegment 1 to

DaySegment 2 (i.e. DaySegment 1 is dropped and DaySegment 2 is added). This observation is the basis for the diagnosis. By differencing two decision trees, one can characterize the change. This algorithm is sufficiently efficient because it stops on each branch comparison when it finds the first difference (or leaves) [29]. There are, however, some issues with differencing decision trees.

3.2.2. Significant Attributes in Decision Trees

In general, completely comparing unordered trees is computationally expensive (NP-complete) [43] Even if I assume ordered trees, which is common for decision trees, the algorithmic complexity of complete comparison is high. The complexity arises because a node in tree A may move to anywhere within the next tree B (however unlikely in practice). (In practice, I have found that the comparing sequential decision trees reveal small changes in the tree, such as the attribute or attribute value changes.) Thus, the entire tree must be exhaustively searched. Even so, complete comparison is practical for small decision trees. Yet, I do not need a complete comparison for the diagnostic problem. Instead, an algorithm that returns the most influential changes is sufficient.

Decision trees use an attribute selection measure as a criterion for splitting at a node. Popular choices include information gain, Gini impurity measure, and gain ratio. The attribute with the highest metric determines the splitting rule. The improvement at each node is calculated using the selection measure and fraction of data split. For each attribute, the importance measure is derived using a weighted sum of the improvements due to that attribute. Such importance values can be used to rank the attributes for a given tree.

More influential attributes are found at the top of the tree, while less influential attributes are found closer to the leaves [44]. Thus, when comparing two trees, the most significant differences will be found by first comparing the roots and working towards the leaves. I apply this method, stopping along each branch when either a difference or a leaf-node is found [29].

By comparing structures of two trees based on the order in which attributes appear near the top, I can determine the differences in influence of an attribute between two models. This ordering of attributes by significance simplifies comparison and is one reason why I use decision trees to model behavior.

Given two decision trees, there are two kinds of differences to consider: (1) value differences at the nodes, and (2) attribute difference. Additionally, it is possible that the two models have nothing in common. These issues are considered next.

3.2.3. Model Differences in Attribute Values

Given two decision tree models, it is possible that they have an attribute value difference. As an illustration, consider the models in Figure 9 and Figure 10. The two trees branch on day segment. The absence of a branch for DaySegment =2 in Figure 10 is the only structural difference.

Dropping a tree branch, from m_1 to m_2, occurs when the latter model is constructed from a data set lacking data for the branch. In the case of Figure 10 compared to the prior model of Figure 9, the user quit reading email during DaySegment =2. (The DaySegment attribute represents a range of hours in the day, for example, from 2 A.M. to 4 A.M.)

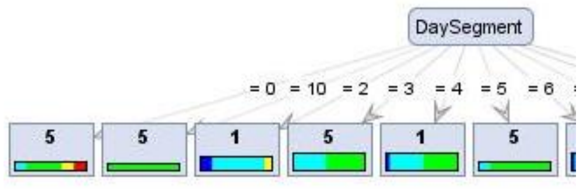


Figure 9 *A decision tree that classifies DaySegment =2*

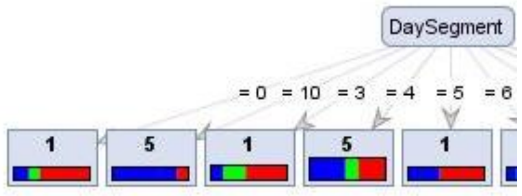


Figure 10 *A decision tree that lacks DaySegment =2*

3.2.4. Model Differences in Attributes

Given two decision tree models, it is possible that they have an attribute difference. As an illustration, compare Figure 9 with the previous Figure 108. The attribute difference occurs in the path DaySegment =4. In Figure 9, there is no subsequent branching, whereas in Figure 8 DaySegment =4 is segmented into three weekday branches (1, 6, and 7).

Adding an attribute, from m_1 to m_2, occurs when the latter model is constructed from a data set that includes data for the branches (e.g., weekdays 1, 6, and 7), and that data is non-uniformly distributed over the attribute (weekday). Thus, I can infer from Figure 8 that the user, during DaySegment =4, mostly had email events on weekdays 1, 6, and 7.

I cannot infer specific quantity changes from attribute differences alone—the addition of weekday in Figure 8 does not suggest that more events occurred during weekdays 1, 6, and 7. Instead, it shows that the distribution of events changed from Figure 9 to Figure 8. It is likely, however, that fewer events occurred for the weekdays other than 1, 6, and 7, thereby creating a non-uniform distribution.

It is worth noting here the pathological case of two identical models, m_1 to m_2, for which the events occur with the same distribution, but different quantities. For example, consider Figure 9. There is no indication of the quantity of DaySegment =10; the figure simply indicates that at least one event occurred at DaySegment =10. It is possible, but not likely, that another model, identical to that of Figure 9 would have twice as many total events with the same distribution. Therefore, the decision-tree model differencing is a diagnostic aid, which can be improved by computing the classified quantities in each leaf. Once a tree difference is identified, the algorithm does compute the difference in data counts. However, analysis is not applied to identical trees, because the pathological case is exceedingly unlikely in practice.

3.2.5. Models with No Common Attributes

Given two decision tree models, it is possible that they have no attributes in common. For example, model m_1 branches only on DaySegment 1 and model m_2 branches only on weekday 1. From this example, I can deduce that the concentration of email events was first on DaySegment 1, uniformly distributed for all weekdays, while the second model concentration shifted to weekday 1 uniformly distributed on all hours. Such a dramatic change has not occurred in the data, but may occur in other domains. In such cases, this algorithm reports the entire two trees as the differences.

3.2.5. Four Experiments

This section has introduced the methodology for event stream analysis: data is acquired, streamed to the tool, which then recognizes and diagnoses transitions. The following four sections, show four experiments of applying the methodology, each demonstrating different techniques from the framework within the domain of patient monitoring. The subsequent chapter 4 demonstrates applying the framework to fraud detection.

3.3. Analysis of the Data-Model Stream Mining Approach

I have applied a design science approach to the analysis of my analytic techniques [45]. In particular, I have applied function, integration, and exploratory testing to verify that the algorithms have been implemented correctly, as well as performance analysis to ensure its usefulness in practice [46]. The approach is reasonably efficient, processing an average of 569.5 events every second on Windows Server 2008 using a virtual image of Windows 7 computer with 3.33 GHz processor and 1 GB memory. I am more concerned with its effectiveness at identifying and diagnosing goal transitions. Here I summarize a case study, as applied to a randomly selected patient from the TAL project. The approach identifies four important goal transitions.

3.3.1. Emailing Case Study

In TAL, the email software is personalized uniquely to each individual, as required by his or her treatment plan. Here, I focus on Don. His goals were introduced in section 3.1.1. Reading and composing email are important events in analyzing Don’s behavior. These two events are shown (or implied) in the leaves of Figure 7. Two data mined models, among others, will be continuously analyzed for Don: (1) read email and (2) compose and send email. The classification problem in support of Don’s behavioral analysis can be summarized as follows.

- Input: event type, (email) word count, hour, day of week, week of year, month of year
- Predict: event type, (email) word count, hour, day of week

To prepare for data mining, I follow the classic steps, which are summarized in the following two subsections [47]. Once the data models are generated, I can begin my analysis for recognizing and diagnosing goal transitions, which is described and analyzed in the subsequent subsections.

3.3.2. Data Logging and Transformation

The TAL project provides an automated custom logger, which appends to a text file. To obtain real-time data access, the log file can be monitored with changes sent to a server. Here is a simplified entry from the log file.

```
09:48:41 NewMailEvent [id=765406159;in-reply-to=311149530;chars=770;words=179;sentences=16]
```

This logged event specifies the time, the program event, and its associated arguments. The example logs the arrival of a new email that is in reply to previous e-mail; the identity of the sender and receiver and characteristics of the e-mail message, such as its length, are also included. The log entries serve as an index into the database of email messages and related (manually entered) information.

To simplify the processing of the data, the event log is transformed into a standard format, Common Base Event (CBE) before it is persisted to an SQL database. To support data mining I do provide a variety of computed fields, such as date views for hour of day, day of week, day of year, etc. The entire process can execute in real-time as each event is generated from the TAL system.

3.3.3. Data Selection and Mining

With the aim to monitor a patient's hourly behavior, I specify a data modeling scheme to analyze a patient's events at each hour. I partition the data based on a patient ID and event type. Additionally, I select the fields for data mining. These include hour, day of week, and word count. The word count field (for compose events) is an integer; to increase predictability (of word count), it is mapped to enumerated sets: {0 – 49}, {50 – 99}, {100, 499}, {500, 999}, {1000, 1999}, {2000, 2999}, {3000, 4999}, {5000, 6999}, {7000, 9000}.

My data mining procedure applies classification to sliding windows of the selected data. In pre-testing, I considered a variety of classification models, including neural networks, Bayesian networks, decision trees, and association rules. I settled on three decision tree algorithms.

- M1: Gain Ratio (C4.5) is a successor of ID3.
- M2: Information Gain (ID3) minimizes the information needed to classify the data represented as tuples and the resulting partitions reflect the least randomness or impurity in these partitions.
- M3: Gini Index (CART) uses a formula based on probability to branch on nodes.

In pre-testing, I also considered a variety of window sizes. I settled on running three window sizes simultaneously: two-, four-, and eight-week windows. My analysis of one patient (Don) included 3,695,086 records occupying 737 MB in Microsoft SQL 2005. The data is automatically processed by custom extensions to Rapid Miner (http://rapid-i.com/). My stream-miner plug-in differs from the standard plug-in in that it: (1) divides data by date and time (e.g., 2-week windows) in addition to a count of datum (e.g., 100); (2) generates a log of information about the models it creates, including precision, accuracy, and other metrics, and (3) outputs the models for subsequent processing, such as model differencing.

3.3.4. Recognizing Goal Transitions

This application of data mining seeks to find interesting changes in behavior rather than rules of behavior. Thus, I look for inflection points in predictive quality, as described in section 1.3.3.

The models predict for a time-slot within a day-of-week. Table 3 summarizes accuracy and precision for the read and compose event types. The values for the different window sizes are very similar. Notice that the read models are very good—meaning that the models predict well the number of emails that will be read during a time-segment on a day-of-week. The compose model is useful but not as good. It weakly predicts the number of emails that will be composed during a time-segment on a day-of-week. Because read models have the best quality and least variability, I rely on them to predict behavioral changes. In particular, the read-model is reliable, and thus can predict changes. The compose-model is less reliable and thus its compose predications can be useful but with greater variance. Therefore, the software first use the read model to raise the a goal transition alarm, which initiates diagnosis. The compose graph mostly supports the same transition points. (See 4, which compares the two model qualities.)

Consider the read model alone. The predictive qualities of the three data mined models (M_1, M_2, M_3) for three windows are graphed in Figure 11. Of course, it's difficult to see the details in the graph, but the overview is useful. The graph shows that the models roughly track the same events. Windows of four and eight weeks are nearly the same. They both the miss or diminish events considered interesting by the two-week window (e.g., weeks 66 – 68). In general, the two-week window analysis has greater variability and foreshadows the larger window analysis, as expected. Notice that week 34 has poor predictive quality (q) as identified

Table 3 *Qualities of model M1 for read and compose email*

		Read		Compose	
		Accuracy	Precision	Accuracy	Precision
2-week window	Average	0.967	0.970	0.631	0.484
	StdD.	0.049	0.018	0.241	0.144
4-week window	Average	0.984	0.971	0.657	0.478
	StdD.	0.027	0.011	0.202	0.119
8-week window	Average	0.983	0.971	0.657	0.478
	StdD.	0.027	0.011	0.202	0.119

in both the two- and four-week window analysis. Moreover, the quality in the immediately surrounding weeks is good; thus, the rate in change of the quality (q') around week 34 is also high. Together the dramatic decrease in q and zeroing of q' suggest that week 34 may be a behavioral inflection point worthy of further analysis.

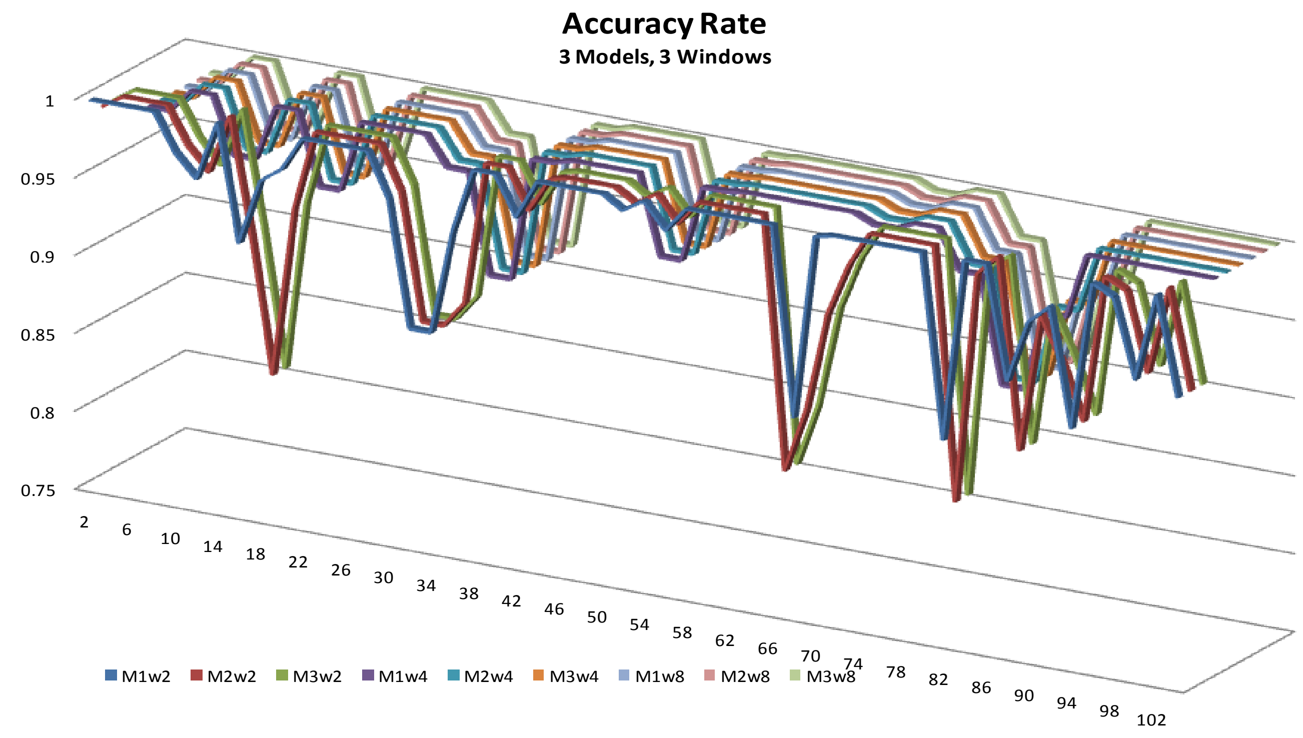


Figure 11 *Predictive quality for the read models (x-axis weeks, y-axis percent)*

Table 4 presents q' for two- and four-week windows for the three models; it illustrates the significance of my transition identification technique. The table shows where q' turns negative between 0 and 36 weeks. The highlighted cells indicate where the value is less than one-half of the standard deviation for values in the column. The analyst can combine these values—for example, using average, logical And or logical Or—to automate the recognition of interesting events. In my case, the software uses logical Or—if any q' turns negative, then that week (or consecutive weeks) is a potential behavior transition and diagnostic analysis is subsequently applied.

Table 4 *First derivative of accuracy, q*

	2-week window			4-week window		
Week	M1	M2	M3	M1	M2	M3
10	-0.012	-0.012	-0.012	0.000	0.000	0.000
12	-0.007	-0.007	-0.007	-0.018	-0.018	-0.018
16	-0.037	-0.037	-0.037	0.018	0.018	0.018
18	0.021	-0.043	-0.043	0.000	0.000	0.000
20	0.005	0.054	0.054	-0.023	-0.023	-0.023
30	-0.014	-0.014	-0.014	0.000	0.000	0.000
32	-0.040	-0.040	-0.040	-0.010	-0.010	-0.010
36	0.033	0.009	0.009	-0.032	-0.032	-0.032

For each goal transitions identified, goal analysis is automatically applied. This analysis for Figure 11 is presented next.

3.3.5. Diagnosing Goal Transitions

As goal transitions are identified, diagnosis is automatically applied. For example, if a goal transition is identified in the mined read model, and then diagnosis will be applied to the read model as well as other models such as the compose email and delete email models. This is based on the assumption that a significant change in any usage of the AT may signal a goal transition for any of the monitored goals. Thus, identified goal transitions are considered signs of behavioral changes that may be reflected across the AT system usage.

Table 5 and Table 6 summarize the diagnostic analysis for the goal transitions identified in the read email and compose and send email data streams underling Figure 11. I present the results of comparing pairs of M_3 (CART) decision trees over 2-week windows. However, the results are similar for the other models and windows.

Table 5 presents the results of model differencing the inflection points found in the compose and send email data stream. One potential goal transition occurs during the two weeks of 32 and 33 of the data set. The three data windows for this inflection point begin at 32, 34, and 36 weeks. The model differencing technique compares the models beginning at 32 and 36—it does not consider the intervening transitional model.

The results are shown in Table 5. For the transition at week 32, there was a 15% increase is email composition during the period from 6 to 12 A.M. and 2 to 8 P.M. There were 48% and 9% increases for the inflection points of beginning at weeks 50 and 80, respectively.

Table 5 *Significant changes in composing email*

Weeks	Compose in selected hours	Total Compose for Model	% Activity	Δ Activity
10 PM				
10 - 12	7	111	6%	
14 - 16	0	115	0%	-6%
Tuesday (6-12 AM and 2-8 PM)				
32-34	3	46	6%	
36-38	14	67	21%	15%
Saturday				
50-52	0	55	0%	

54-56	39	81	48%	48%
Saturday, Sunday, Monday and Friday (6-12 AM)				
80-82	5	31	16%	
84-86	23	91	25%	9%

Table 6 presents the results of the model differencing for inflection points found in the read email. For the transitions beginning at weeks 32, 50, and 80 the increases were 1.2%, 21%, and 3%, respectively.

Table 6 *Significant changes in reading email*

Weeks	Read in selected hours	Total Read for Model	% Activity	Δ Activity
10 PM				
10 – 12	6	290	2%	
14 – 16	0	365	0%	-2%
12 AM - 2 AM				
32-34	0	146	0%	
36-38	2	162	1.23%	1.23%
Sunday				
50-52	0	203	0%	
54-56	57	272	21%	21%
10 PM – 2 AM				
80-82	0	121	0%	
84-86	13	442	3%	3%

The preceding tables summarize how model differencing applies. The tables demonstrate how transitions can be characterized by their underlying behavioral changes. It should be noted that, for this presentation, I collapsed consecutive time segments. The actual mined model considered 2-hour day segments. Thus, for example, the Saturday statistics of Table 6 is an aggregation of the 12 time-periods. The other statistics are automatically calculated (e.g., total and percent). Finally, it should be noted, that the results are calculated in real-time, as each window closes, immediately after the calculation of goal transitions.

The TAL clinicians use the diagnostic results to evaluate patient behavior. The read and compose email data mined models guide the evaluation of Don's lower-level goals, illustrated in Figure 7. Logical property-based analysis is used to evaluate the satisfaction of achievement goals, like after receiving an email, a user shall read and reply to the sender, within k days [30] [48] [49]. Analysis of the preceding data mined models provides feedback on goal trajectories, e.g., the user is increasing email composition. Review of the goal transition diagnosis quickly directs clinicians to the most important behavioral changes. These changes may not include the anticipated increase or decrease of a goal, like email composition. Instead, the changes can include other attributes, such as time, day of week, etc. The preceding Table 5 and Table 6 show an increase in weekend behavior and some decrease in late evening (10 P.M.) behavior. Such unanticipated diagnoses have been helpful in understanding the total changes a patient is undergoing. In Don's case, he had less AT usage during traditional sleep times—a sign of improvement.

3.3.6. Evaluation

My research evaluation considers two concerns, introduced at the beginning of this research: Can I provide automated recognition of changes in patient behavior, where a patient's behavior is defined by the stream of events that they initiate with the AT?

Can I provide automated diagnosis of a patient's behavioral change, as characterized by the most prominent behavioral differences around a moment of change?

Through software construction, testing, experimentation, and case study, I affirm both propositions. Next, I consider two more issues: (1) model quality and (2) validation by real-world events.

Accuracy and Precision

Accuracy and precision are standard metrics for determining predictive model quality. As Table 3 shows, the read model is very good and while the compose model is useful not as good. This may be an anticipated because the read model depends on received emails and free time of the patient; these elements are mostly routinized for the user population. On the other hand, email composition depends on the patient's skills and interest in communicating. Such interest depends on the content of the received email as well as external, non-modeled events, such as the correct usage of medicine or the arrival of visitors. In a prior study, I showed that patient interest and email composition increased with the addition of new email buddies, while decreasing slowly thereafter [30]. Therefore, given the limited information in the data available and the real-world behavioral variations of the users, the availability of at least one very good model appears to provide adequate information to identify some significant behavioral changes.

Real-World Events

Changes in model quality reveal changes in user behavior. My case study illustrates this with the inflection points that Table 5 and Table 6 summarize. Note that there is a causality chain from the user's manipulation of the AT to the diagnosis of goal transitions:

- A user exercises the AT interface, such as reading and composing email.
- Data mined models are generalized from and accordance with the distribution of the events.
- Model differences are calculated, revealing changes in the models, which reflect changes in the event distribution, and thus changes in the user behavior.
- Model differences, at goal transitions, are characterized according to the events observed.

Thus, assuming the algorithms and software are correct, there is a direct causal chain from changes in usage of the AT and the diagnosed changes presented to clinicians.

I discussed the results with the other TAL researchers. The goal transitions do seem to reflect persistent changes in behavior. One example may be revealing. Week 34 of the data corresponds to 8/20/2006 - 8/26/2006, while week 82 of the data corresponds to 7/29/2007 - 8/4/2007 (week 31 of 2007). I hypothesize that something interesting happens to the patient in the August summer holiday, such as a family member visit. Patient anonymity prevented me from directly correlating such real-world events, but it has been intimated that such events have occurred.

Threats to Validity

I believe the design science objective of extending existing theory and creating new technology has been met [50]. Through testing, performance analysis, and case studies I have demonstrated the effectiveness of the techniques for discovery and diagnosis of goal transitions.

A review of the related research supports the novelty of using model quality and model differencing of stream-mined models. The main validity concerns are the accuracy of goal transition identification and generalizablity of the approach, which of course, is a ubiquitous research concern.

Transition identification can be tuned using parameters of model quality. The analyst can select the metrics (e.g., accuracy, precision) that specify the quality metric, q, as well as how multiple models qualities are compared against the threshold to indicate a potential transition. The following illustrates the rule that if two models, with an intervening model, have a negative change in quality then consider the intervening model a transitional model:

If ($m_i.q' < \varepsilon$ and $m_{i+2}.q' < \varepsilon$) then consider m_{i+1} a potential transitional model

The analyst must tune the transition identification parameters. Too many potential transitions will consume too much effort from the clinicians, who use the results to evaluate the clinical plan. Too few potential transitions will result is missed opportunities to adapt the AT or train the user. Therefore, the analyst must work with the clinicians to tune software to an appropriate threshold. In the TAL research, clinicians find the notification level reasonable and useful in their work of monitoring dozens of patients.

Generalizablity of the approach is another scientific concern. Model quality and model differencing is appropriate for patient behavioral monitoring, as demonstrated here. That alone is a significant contribution. However, can model quality and model differencing be applied effectively to other domains, such as business activity monitoring or network analysis? This open question has provided me a new research opportunity in the area of financial business activity monitoring.

Consider the problem of monitoring personal credit card transactions, which is similar to the problem I am undertaking. Although simpler than rehabilitation goals, users do have goals for using credit cards, such as short-term payment, long-term payment, and insured (rental car) transaction. Financial models can be stream mined from transactional data. Behavioral changes can be inferred from model quality changes and characterize via model differences. Consequently, as a user transitions from short-term financing to long-term financing, the techniques described herein can automatically flag and characterize the change—at least my initial efforts suggest that this may be appropriate. It’s fair to say that analyses of stream-mined models, such as quality metrics and differencing, provides research opportunities for real-time data analytics.

3.3.7. Experiment Conclusion

Section 3.3 presented technique for monitoring and diagnosing behavioral changes from an event stream of user actions. This problem arises in the context of applying AT to aid cognitive treatment plans. Users work towards long-term goal achievement by applying planned behaviors.

To analyze user behavior, I analyze the quality of models generated, in sequence, by mining the stream of user events. Changes in quality suggest significant changes in user behavior. These goal transitions can suggest the need to change a user’s treatment plan, moving forward with plan achievement and revisiting prior treatments with user regression. My case study of use demonstrates the feasibility of this approach. The entire process is automated—from event acquisition in the AT software to recognizing and diagnosing changes in model quality.

3.4. Cognition and Sequencing

Goals are satisfied by task sequences. Planning and executing complex task sequences entails greater cognitive loads than simple sequences. Task complexity can be viewed as: (1) primarily a psychological experience, (2) an interaction between task and person characteristics, and (3) a function of objective task characteristics [51]. In each view, sequencing of tasks increases complexity, especially if it includes alternative tasks and subsequences [52]. The ability to think about task sequences that are lengthy and variable is an aspect of good cognitive processing [53]. Unvarying short task sequences are considered less indicative of good cognition than moderate to long, varied and repeated sequences.

Recognition and differential diagnosis of action sequences is the primary contribution of this research section. My monitoring software records and analyzes task sequences initiated by the client. In particular, the software finds repeated patterns, or motifs, in the UI event stream. Motif analysis provides caregivers insight into cognitive effort that a client devotes to emailing. A smaller effort suggests a lack of interest, software obstacles, or troubles in cognitive processing. Trend analysis is most important. Clinicians consider waning motifs as a trouble sign, whereas motifs that are more complex suggest improvement in interest, UI configuration, or client cognition. Interesting UI event motifs suggests that the client has mastered emailing, and thus their *Activities of Daily Living* (ADLs) will improve through improved communications.

3.4.1. Stream Mining Sequences

To recognize and diagnose sequences, I apply sequence mining logged UI actions. My software identifies usage sequences and supports trend analysis of usage sequences.

Events occur in sequence. A sequence pattern specifies a sequence of events that frequently occurs within a period. Finding patterns of reoccurring sequences is the goal of sequence mining [54].

Sequence mining can be applied to a series of data windows using an approach called stream mining. Stream mining systems analyze voluminous, continuous data streams where it is not practical to store all the data. Instead, sequential data subsets, called windows, are analyzed as they arrive. This is the approach I apply—sequence mining of data windows and trend analysis across the windows.

Such analysis is useful to clinicians, as they monitor their clients. There had been no knowledge of UI sequences in the TAL project. With the stream sequence-mining described herein, clinicians now gain insight into client UI trends, which they may associate with changes in client interest, software obstacles, or troubles in cognitive processing.

Process Mining

Process mining extracts information from event logs to derive business-process descriptions [55]. Constructing a business process model from a transaction log is a typical application. Inferring casual relationships, including concurrency, loops, conditionals, task hierarchies, etc., of the task network from time-stamped logs in the face of noisy and incomplete (erroneous) logs of process events is the focus. Nascent research areas address noisy and incomplete logs, visualizing results, and delta analysis, which explain differences between two derived models.

In many respects, process mining is related to AT mining. A significant difference may be that concurrency is more common in business processing than the pursuit of concurrent goals by a human. Moreover, a human is more likely to be inconsistent when pursuing goals, as well as opportunistic in pursuing new goals. Thus, human event logs are less uniform and therefore more weakly structured than business process logs.

Sequence Mining

Sequence data mining concerns analysis of events in sequence. The event data are often nominal-valued or symbolic and the goal is to discover variables and their correlations [54] [56]. This contrasts to the well-studied domain of time series analysis, which considers real or complex-valued time series of known parameters using methods such as autoregressive integrated moving average (ARIMA) modeling.

Sequence mining techniques address: (1) prediction, (2) classification, (3) clustering, (4) search and retrieval, and (5) pattern discovery. My task of sequential pattern discovery naturally leads to the consideration of motif analysis [57]. Work in this area discovers patterns from a database of protein sequences [58], for example. Similar work finds frequently reoccurring patterns [30], and some address explicit time constraints [59].

I apply the Max Motif algorithm, which efficiently enumerates all maximal motifs in an input string [60]. Algorithm parameters consider the minimum occurrence threshold for pattern consideration, and the maximum of variables (aka wildcards) within the pattern. For example, B*AB**B is a pattern of length 7 with max wildcard length of 2. If the minimum occurrence threshold is 3, then this pattern will only be reported if it occurs 3 or more times in the input. The algorithm only reports on maximal motifs, which is a motif that is not properly contained by other motifs.

3.4.2. Max Motif Sequence Mining

I use KNIME for data mining (www.knime.org) Figure 12 shows a KNIME workflow for applying Max Motif in batch mode. (The same operators apply in real-time.) The Database Reader, at the left, identifies the data windows that are retrieved by the Database Reader at the lower left in Figure 12. Each window is processed within the loop by the Max Motif operator.

Time is an important concern for the TAL event sequences. Consider the following TAL email-client events:

1. NewMailEvent
2. ReadEmailEvent
3. ComposeEmailEvent

If the events all occur within a few minutes of each other, then they may be interpreted as a user creating (and sending) an email in response to a newly arrived and read email. Such a sequence is considered an important pattern. On the other hand, if there is a 6-hour gap between the Read and Compose events, then it seems less likely that the user is responding to the newly arrived email. Consequently, a time-gap splits a sequence into two conceptually significant sequences.

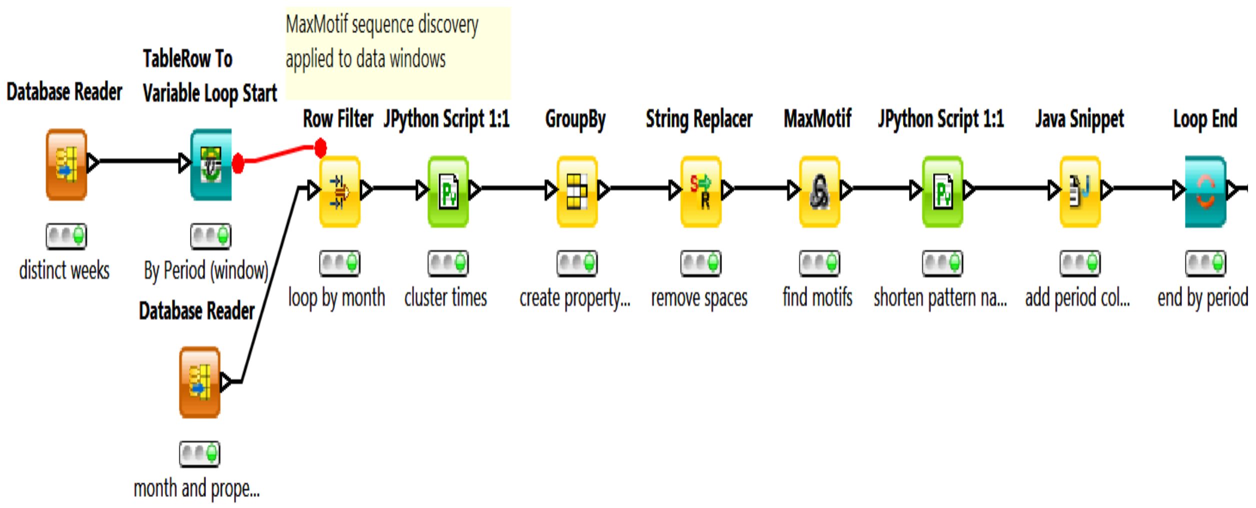


Figure 12 *KNIME workflow for applying Max Motif to data windows*

To address time gaps, I pre-process the input events. The leftmost JPython Script in Figure 12 segments sequences, ensuring that there is no more than a 2-hour gap between events. This approach allows me to address time constraints prior to applying Max Motif, which efficiently discovers sequence patterns without regard to time.

The remainder operators in the KNIME workflow of Figure 12 address data representation: (1) encoding and decoding the TAL events names for Max Motif, and (2) adding window information to the data record.

In support of trend analysis, the software determines differences in pattern sets, as discovered in consecutive windows. Consider the patterns discovered in window w_i as model m_i. I apply model differencing to discover pattern changes from window w_i to wi+1: $(m_{i+1} – m_i)$. Figure 13 shows how pattern differencing is applied within KNIME. The JPython Script in Figure 13 calculates differences for the consecutive models produced by the Max Motif workflow of Figure 12. The Java Snippet and Value Counter operators calculate aggregate pattern information, including the number of wild cards, the total number of patterns identified within a window and within all windows, etc. The results of these analyses are presented as pivot tables and graphs.

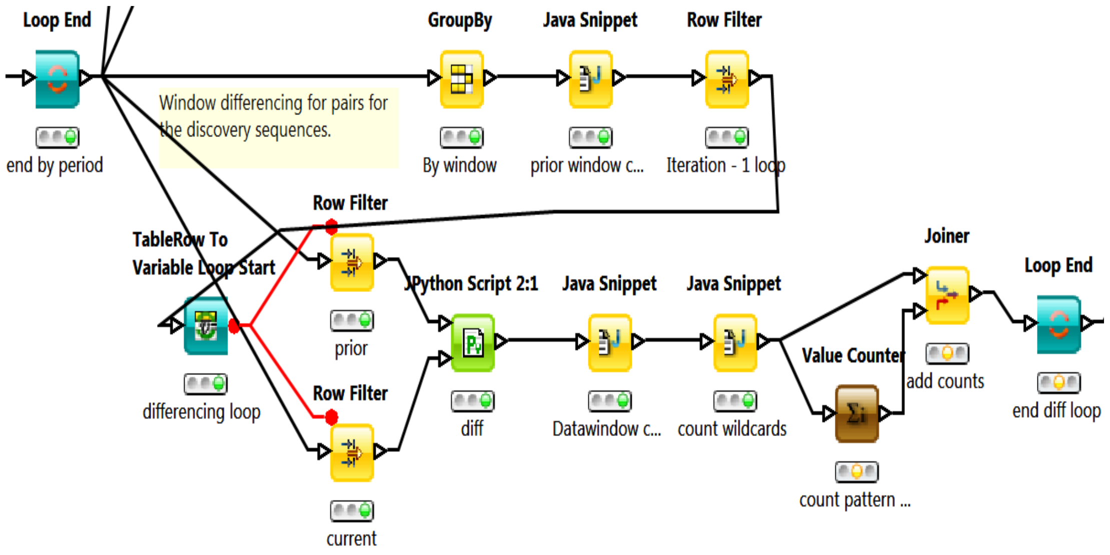


Figure 13 *KNIME workflow for differencing sequence patterns between data windows*

3.4.3. Emailing Case Study

TAL Data Stream

The TAL email client provides an automated custom logger. To obtain real-time data access, a log file can be monitored. Here is a simplified entry from the log.

```
09:48:41 NewMailEvent [id=765406159;in-reply-to=311149530;chars=770;words=179;sentences=16]
```

This logged event specifies the time, the program event, and its associated arguments. The example logs the arrival of a new email that is in reply to previous e-mail; the identity of the sender and receiver and characteristics of the e-mail message, such as its length, are also included. The significant event types are: read email, compose email, delete email, and new (arriving) email. A database view provides a continuously updated stream. The dataset for one client, Don, included 3,695,086 records occupying 737 MB in Microsoft SQL Server 2005. These data are assumed for the remaining discussion.

Preprocessing and Mining

Rather than data mine the raw TAL event stream, I pre-process the data using my property-based monitoring system, EEAT. This system allows me to encode temporal relationships as properties. For example, a response property can be defined as A followed by B within 6 hours.

The Event Engineering and Analysis Toolkit (EEAT) supports specifying, instrumenting, data collecting, and property analysis [61] [62] [63] [64] [65] [66] [67] [29]. It is a requirements monitoring system in that it supports (1) analysis of abstract, requirements-level properties, and (2) automation of runtime requirements evaluation, which interprets low-level software events as contributors to the eventual satisfaction or violation of requirements [68] The EEAT goal specification language is a variant of the OCL 2.0, which we call OCLTM—meaning OCL with Temporal Message logic [41]. It is similar to other languages that support some form of predicate calculus and temporal logic over an object model (e.g., KAOS [46]), and thus supports Goal Oriented Requirements Engineering (GORE) [69] [70]. OCLTM also includes real-time operators.

```
package talx::model
context TransportToolkit
  def:       timeout1: LTL::Timeout = timeout('0d:6h:0m:0s')
  def:  rReadMail: LTL::OclMessage = receivedMessage('talx.events.ReadMailEvent')
  def:  rComposeMail: LTL::OclMessage = receivedMessage('talx.events.ComposeMailEvent')
  def:  rDeleteMail: LTL::OclMessage = receivedMessage('talx.events.DeletMailEvent')
 -- argument(1) is the buddy associated with the email
  inv emailToBuddy_1: eventually(rComposeMail.argument(1)='-1266262578')
//...
  inv emailToBuddy_31: eventually(rComposeMail.argument(1)='81287508')
  inv readFromBuddy_1: eventually(rReadMail.argument(1)='-1266262578')
//...
  inv readFromBuddy_31: eventually(rReadMail.argument(1)='81287508')
  inv replyToEmail_1: after(eventually(rReadMail.argument(1)='-1266262578'),
         eventually(rComposeMail.argument(7)=rReadMail.argument(6)),timeout1)
//...
  inv replyToEmail_31: after(eventually(rReadMail.argument(1)='81287508'),
         eventually(rComposeMail.argument(7)=rReadMail.argument(6)),timeout1)
endpackage
```

Figure 14 *EEAT OCL expressions of TAL properties*

A typical EEAT scenario, as applied to TAL, is as follows:

1. A software analyst specifies TAL goals as formal properties to allow for automated monitoring.
2. The analyst uses an integrated development environment (IDE) to generate automated monitors from the formalized properties.
3. As the user exercises the TAL client, events travel from TAL to EEAT, which calculates property satisfaction.

Figure 14 shows three kinds of TAL properties:

1. Email to buddy. Each of the 31 buddies is enumerated by their ID. The property is true each time the corresponding buddy is emailed.
2. Read from buddy. The same as email to buddy, but for reading emails.

3. Reply to buddy. A read from and then reply to a buddy that occurs within a specific period (6 hours).

Properties like reply to buddy, are true when the time (and data) constraints are met. Thus, they encode temporal properties. These EEAT properties are then mined for sequences. Using workflows like that shown in Figure 12 and Figure 13, I applied sequence discovery to Don's usage of the TAL email client. The resulting tables about the discovered patterns describe:

- Sequence property patterns (including wildcards) e.g., read_16,*,read_16.
- Property counts for windows, e.g., current = 60, next = 119
- Window count, indicating the number of windows that contained the pattern

Analysis of the resulting patterns is presented next.

Sequence Analysis

Don's 10 most frequently occurring motifs are:

1. read_10
2. read_16
3. read_16,*,read_16
4. read_16,read_16
5. read_9
6. reply_10
7. reply_16
8. reply_16,*,reply_16
9. reply_16,reply_16

10. reply_9

The number indicates the buddy involved, while the '*' indicates a wildcard in which any single event may be substituted. Table 7 shows characteristics of the discovered sequence patterns.

Table 7 *Characteristics of sequence patterns*

Sequence pattern	Min	Avg	Max	StdDev
Non-variable Length	1	3.75	15	3.32
Variable length	1	11.32	20	3.64
Wildcards	0	3.68	9	1.84

Figure 15 graphs the average pattern length over 2 years of 2-week data windows, revealing the slightly decreasing pattern length. Figure 16 graphs non-variable pattern lengths, revealing a slightly increasing length. Figure 17 reveals the patterns by the initiating event, showing three properties from Figure 14.

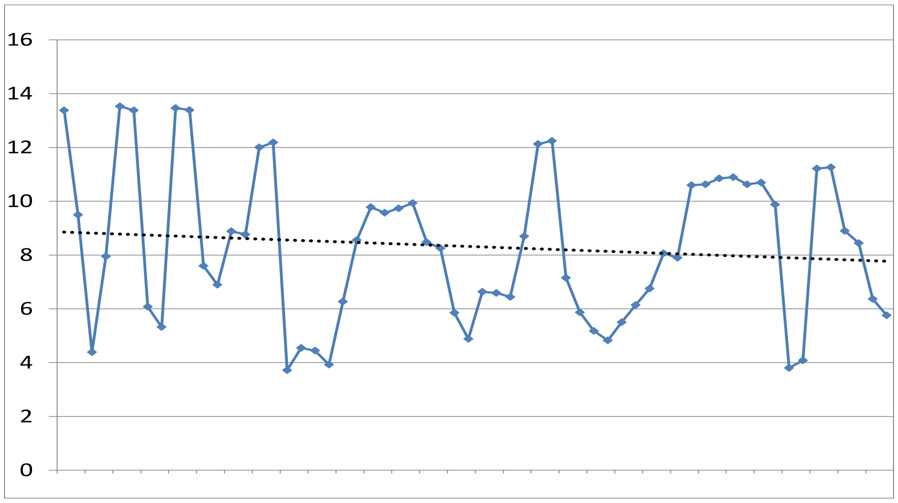


Figure 15 *Average variable pattern length*

The topmost dashed line shows great variability in the increasing length of patterns that begin with read. The middle dotted line shows patterns beginning with compose are more consistent in length, but are slightly decreasing over time. Finally, the bottom-most solid line shows the eventual increase in length of patterns beginning with a reply.

Lastly, Figure 18 shows the average occurrence of patterns beginning with one of the three properties from Figure 14. The topmost dashed line shows patterns beginning with read, the middle dotted line shows patterns beginning with compose, while the bottom-most solid line shows reply. Again, reply sequences are most increasing, read least increasing, and compose is decreasing.

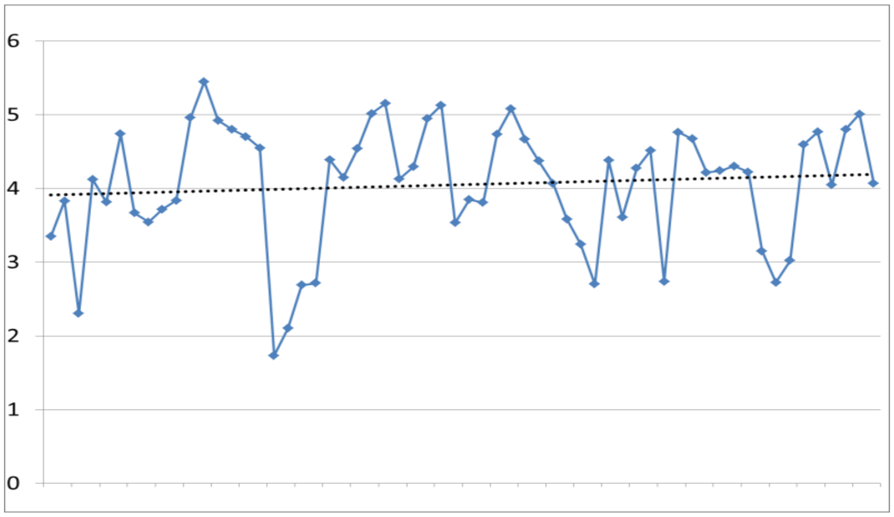


Figure 16 *Average non-variable pattern length*

Overall, this may be interpreted as Don becoming (1) more consistent over time because of the increasing length of non-variable sequences, and (2) becoming more conversational with email because of the increased usage of reply. This latter point is significant, because it suggests that Don may be transitioning from using email for simple notifications (e.g., “I’m low on meds”) or requests (e.g., “I need a ride to the library.”) to dialogs (e.g., “I do remember that about our father. That reminds me of ...”). Prior analysis of simple metrics (e.g., message length, composition speed, etc.) supports this common trend of TAL users becoming increasingly skilled emailers [71].

The dramatic drops in pattern length in the figures are noteworthy. They are consistent with transitions in Don’s emailing behavior, previously identified and diagnosed [28] [72].

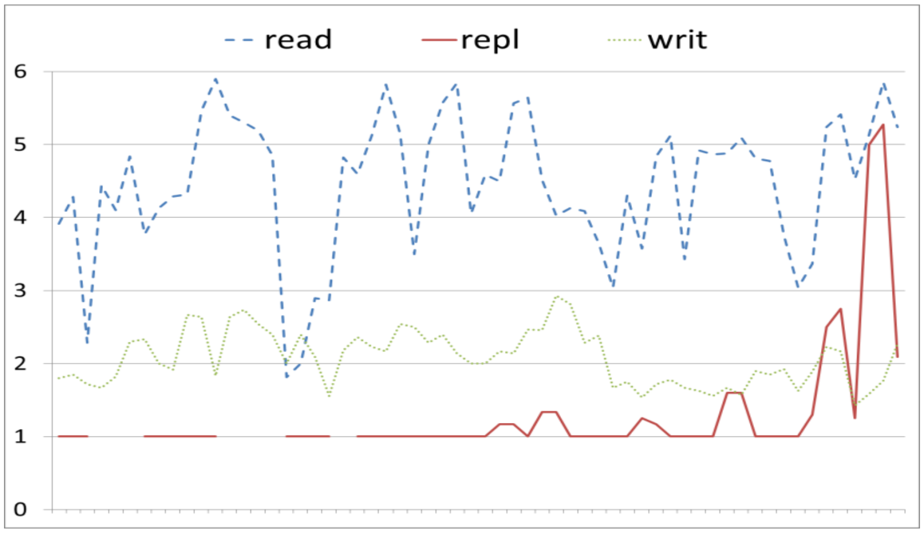


Figure 17 *Average pattern length by first event type*

3.4.4. Evaluation

Analysis of sequence patterns supports clinicians in their efforts to link AT usage with cognitive theories. A domain like emailing allows for planned behavior. A client can read an email and respond. In the context of a discussion, a client may re-read a few prior emails from a buddy and then compose a response. The emailing usage log provides a direct means for observing such planned behavior. Pattern analysis can reveal user plans as variable sequences, even when a user makes occasional mistakes or pursues concurrent goals.

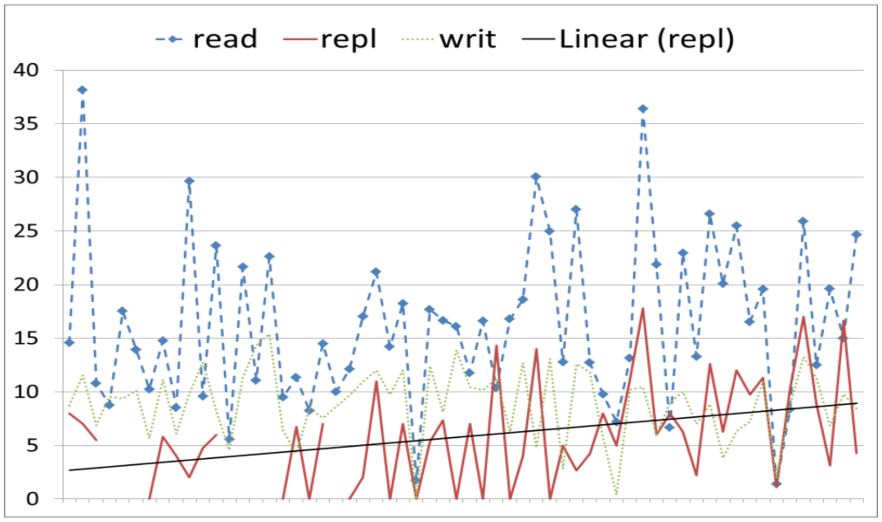


Figure 18 *Average occurrence of non-variable patterns*

The approach presented herein does not consider a statistical analysis of the discovered patterns. It would be helpful to know not just the occurrence counts for the patterns, but the statistical significance of the patterns within the data. It is possible to consider the pattern likelihood against the background of random sequences. This line of reasoning has been explored considering Bernoulli [42] and Markov [73] model assumptions. The next section will consider these concepts in building more complete UI usage models.

3.4.5. Experiment Conclusion

Recognition and differential diagnosis of action sequences is the primary contribution of this section. The monitoring software records and analyzes task sequences initiated by the client. In particular, the software finds repeated patterns, or motifs, in the UI event stream. The software has been applied to monitoring AT usage by clients with cognitive impairments.

The approach appears appropriate for analysis of human event sequences. In particular, it addresses the randomly occurring events that can arise from individuals with cognitive impairments. Parameters of Max Motif can limit pattern detection by their occurrence and randomness (i.e., number of variables). The data windowing of stream mining limits analysis to data segments rather than the whole dataset. These scope limitations focus analysis on localized patterns and their incremental changes. This is the kind of trend analysis needed for monitoring TAL clients.

Clinicians can now consider sequential patterns when assessing their TAL clients. It is likely that the approach applies more broadly to human planed action domains requiring monitoring.

3.5. Transition Identification using HMM

Recognizing change is a core function of monitoring. Consider a stream of UI events divided into data windows, (w_1, w_2, …, w_n). Transition identification marks each data window as either being typical or atypical; for example, (typical, typical, atypical, typical, typical …). Atypical behavior is historically unusual behavior, according to some measure such as statistical variance. I use the term atypical because the behavior is unusual and transient, and thus interesting from a cognitive or learning perspective.

In my approach, the stream of TAL email events is divided into data windows. Each window is characterized by a model, ($w_i \Rightarrow \lambda_i$). Given a transition point, p, that occurs between two models, λ_1 and λ_2, I find the difference of the models to characterize the change: $d\lambda/dt = (\lambda_2 - \lambda_1) / (t_2 - t_1)$, where $t_1 < p < t_2$. In this work, the models (λ_i) are hidden Markov models (HMMs) that characterize a user's behavior. Now, because of this automated differencing technique, the monitoring system can quickly identify modeled changes. Thus, caregivers can review a patient's dashboard to see when a client's behavior changes significantly.

Event sequencing is an important characteristic of user behavior. It's also a challenge for automated monitoring systems to recognize event sequences and notice their changes or trends. Prior monitoring approaches do not address sequences. An approach that relies on decision tree differences cannot directly recognize changes in sequence distributions. A plan recognition approach does address the sequences found in its pre-specified plans. It's good for strict compliance checking; however, it ignores unmatched sequences, and thus is of limited use where there is a great variety of sequential patterns. Thus, I apply a HMM approach, which builds a stochastic model of all sequential patterns. Thus, my approach can characterize sequences in user behavior efficiently. This is a unique contribution of this research.

3.5.1. Learning & Cognitive Rehabilitation

Transtheortical theory (TTM) is a popular stage model to explain behavioral changes. Developed by Prochaska in 1977 by analyzing theories in psychotherapy and behavior change [50], it was first used by Prochaska and his colleges to model self-change and therapy change of individuals of their smoking behavior [45]. In this model, individual's change is defined as a "process involving progress through a series of stages", from precontemplation to action and maintenance [74]. This theory specifies learning states and their transitions, including relapse and recycling back to earlier stages. This state-transition view of learning is still common is more recent theories (e.g., [75]).

Cognitive rehabilitation theory also considers a staged approach, but with explicit goals satisfied by task sequences. Planning and executing complex task sequences entails greater cognitive loads than simple sequences. Task complexity can be viewed as: (1) primarily a psychological experience, (2) an interaction between task and person characteristics, and (3) a function of objective task characteristics [51]. In each view, sequencing of tasks increases complexity, especially if it includes alternative tasks and subsequences [52]. The ability to think about task sequences that are lengthy and variable is an aspect of good cognitive processing [53]. Random and short task sequences are considered less indicative of good cognition than moderate to long, moderately varied (i.e., parameterized) and repeated sequences.

With regard to the emailing domain, a smaller cognitive effort by a client suggests a lack of interest, software obstacles, or troubles in cognitive processing. Trend analysis is most important. Significant changes in behavior suggest that the client is either transitioning, to a more knowledgeable (learned) state or relapsing to a prior state.

The monitoring software records and analyzes task sequences initiated by the client. In particular, the software models sequential patterns and the probability that they fit into the current model. Additionally, the sequentially obtained models are compared to determine changes over time. This approach recognizes:

1. Typical behavior as common, routinized, sequential patterns within a data window.
2. Typical behavior as less common behaviors within a data window.
3. Transitions from one model of typical behavior to a new model of typical behavior (i.e., the new typical). This may be transitioning to a more knowledgeable state or relapsing to a prior state.

To automatically recognize these behavioral forms, I apply sequence modeling within a data stream mining approach.

3.5.2. Sequence Stream-Mining using HMM

Sequence data mining concerns analysis of events in sequence. The event data are often nominal-valued or symbolic and the goal is to discover variables and their correlations [54] [56]. This contrasts to the well-studied domain of time series analysis, which considers real or complex-valued time series of known parameters using methods such as autoregressive integrated moving average (ARIMA) modeling. Sequence mining techniques address: (1) prediction, (2) classification, (3) clustering, (4) search and retrieval, and (5) pattern discovery. I apply hidden Markov models to discover sequences.

I apply sequence mining in the context of stream mining. Stream mining aims to find interesting relationships over a sequence of data segments [47] [76] [77]. Stream mining algorithms can vary substantially from their more traditional forms: data may be analyzed incrementally rather than as a batch, old data may be discounted or removed in favor of newer data, the created model may be an approximation when compared to its traditional form[49]. A variety of techniques can be applied to stream data [48] [78] [79] [80]–much of the work is focused on the efficiency of incrementally updating the model [81].

Detecting changes in data-streams is important for monitoring. Two types of algorithms are common: (1) distribution detection, which watches for changes in the data distributions, and (2) burst detection, which watches for sudden large and unusual changes in a data-stream. Distribution detection algorithms have two common forms: (a) data from two windows (current and reference) are compared using some distance measure, (b) a predictive model is created from a prior window and then its prediction is compared with the current window—high prediction error indicates a significant change. I demonstrate both technique using HMMs.

3.5.3. Requirements Monitoring Tool Support

Runtime monitoring of software for specified behaviors is increasingly of importance to software engineering [68]. Computer published a toolkit for monitoring user-interface activities [82]. More generally, runtime analysis of properties is a growing trend in software development [83]. Goal monitoring integrates and generalizes prior monitoring technologies in support of high-level user or system requirements. A recent article summarizes this research and presents systems demonstrating a variety of concerns and techniques that influence the interpretation of a running system [68].

Table 8 summarizes closely related tool support for behavioral monitoring, mostly from the RE and associated literature. The first column presents the concepts, which are defined in the second column, while the third column presents representative references.

Most work in requirements monitoring checks for runtime compliance with design-time properties. I call this compliance checking because the properties imply or explicitly specify a sequence of actions that shall be executed by the software.

Table 8 *Tool support for behavior monitoring*

#	Concept	Definition	References
1	Compliance checking	Identify absent events compared to specified event sequence.	Deviation from sequences implied by plan—see survey: [68]; Deviation from event-stream properties (e.g., sequences, data predicates, etc.) [27] [84] [11] [85] [40] [62] [64] [86] [87]
2	Aggregate behavior transition identification	Identify a change in event distribution compared to a model of past events.	Decision tree differencing [28][20]
3	Aggregate behavior transition diagnosis	Identify the specific events (types and quantities) that have changed from one observation window to the next.	Decision tree differences and enumerate of associated difference metrics [72] [88]
4	Aggregate sequential behavior transition identification	Identify a change in event-sequence distributions compared to a model of past event sequences.	Simple difference metrics [89]; Markov probability distribution differences (this article)
5	Aggregate sequential behavior transition diagnosis	Identify the specific in event-sequence (types and quantities) that have changed from one observation window to the next.	Markov probability differences enumerate as event-sequence metrics (this article)

Another approach to monitoring analyzes trends of aggregated software events. Rather than enumerating and checking all properties, this approach focuses on finding (row #2) and diagnosing (row #3) unusual events, assuming that most software events are typical. Where there are many potential acceptable event sequences, this approach can be more efficient in specification and monitoring effort. Conversely, where there a few potential acceptable event sequences, the specify-and-check approach of row #1 is more efficient.

The last two rows of Table 8 address the discovery of event sequences. Rows 2 – 5 are only necessary when one wants to monitor and analyze implicitly specified properties (i.e., data mining properties). If (1) all properties of interest can be specified, and (2) the closed world assumption applies, then the compliance checking approach of row #1 is appropriate. If either of these preceding assumptions are inappropriate, then the stochastic approaches of rows 2 – 5 are appropriate. The best approach may be to apply compliance checking for important properties along with stochastic anomaly analysis; together, they increase the likelihood that the monitor will identify interesting software activities.

It is important to consider the unusual characteristics of the events in this domain. Users with CI can engage is some unusual behaviors, which result data with large variance. One client periodically forgot his medication on Wednesdays, and then engaged in the unusual behavior of "drawing", via placement of characters, within the email compose form. The sequence of edits without email sent, as well as the randomness of this episode, calls for monitoring methods that address sequences and probabilities.

Once a change and patterns is identified, the next question that arises is "what exactly did change?" This leads to the research in a row #5. Here, I show how Motif differencing can be applied to the windows associated with the transition. This approach originated in [89] (row #4). That approach only provided simple counts of the sequential patterns discovered. Herein, I show how the probabilities of such patterns can also be obtained.

From the domain of credit card fraud detection, Srivastava et. al. uses HMMs in a manner similar to that presented herein [90]. To detect fraud within a client's transaction stream, a single HMM characterizing past transaction is created. Now, as the client uses her credit card, a window on the growing transaction sequence is incrementally compared with the HMM—a low acceptance probability suggests the presence of fraud. (See equation 8 in the following Section) My work differs in that: (1) I dynamically create HMMs with each data window, and (2) I apply two different techniques to detect distribution changes.

3.5.4. Monitor Design

Hidden Markov Modeling

A hidden Markov model (HMM) is a stochastic signal model [91]. In my application, the signals are sequences of discrete typed events. A HMM provides algorithms to solve three important problems:

1. Compute the probability that an observed sequence, O, is represented by a HMM, λ (using the Forward-Backward Procedure [61]).
2. Adjust the parameters of a HMM, λ, to maximize the fit to an observed sequence, O (using the Baum-Welch algorithm [62]).
3. Compute the optimal HMM state sequence that best explains an observed sequence, O (using the Viterbi Algorithm [63]).

In this presentation, I show how applying the first two algorithms help to identify unusual (atypical) behaviors from an event stream.

A HMM can be characterized as follows:

N is the number of states in the model, $S = \{S_1, S_2, \ldots S_N)$, where S_i is a state. Time t denoted by q_t. **(1)**

M is the number of distinct observation symbols per state; the set is denoted as $V = \{V_1, V_2, \ldots V_M\}$. **(2)**

The state transition probability matrix $A = \{a_{i,j}\}$, where $a_{i,j} = P(q_{t+1}=S_j|q_t=S_i)$, $1\leq i\leq N$, $1\leq j\leq N$; $t = 1, 2, \ldots$ Where any state can reach another state in a single step, we have $a_{i,j}>0$ for all i,j. Additionally, **(3)**

$\sum_{j=1}^{N} a_{i,j} = 1, 1 \leq i \leq N.$

The emission (aka observation symbol) probabilities matrix $B = [b_j(k)]$,

where $b_j(k) = P(V_k, S_j)$, $1\leq j\leq N$, $1\leq k\leq M$ and $\sum_{k=1}^{M} b_j(k) = 1, 1 \leq j \leq$ **(4)**

N

The initial state probability vector $\pi = [\pi_i]$, where $\pi_1 = P(q_1=S_1)$, $1\leq i \leq$ **(5)**

N, such that $\sum_{j=1}^{N} \pi_i = 1$.

The observation sequence $O = O_1, O_2, O_3, \ldots O_R$ where each observation

O_t is from V, and R is the number of observations **(6)**

I use the common notation $\lambda = (A, B, \pi)$ to indicate the complete set of parameters of the model, where A, B implicitly include N and M.

In my HMM application, the symbols of the observation sequence contain two email event types, V = {compose, read}. (An HMM can efficiently handle hundreds of observation types.) The states are the two learning states of the client, S = {typical, atypical}. (HMMs typically have a few states, with about 10 being the maximum to handle efficiently. The states represent contingencies under which different probabilities are observed. Here, I assume two sets of probabilities, based on the user's learning state—either typical learning, or atypical learning.) I use HMM algorithms to construct and compare HMMs representing the stream of observed email events.

Identifying Transitions

Transition identification detects significant changes in modeled events between consecutive windows of event data. HMMs can be used to identify transitions by: (1) comparing consecutive HMMs generated from the observation sequences, or (2) comparing consecutive acceptance probabilities. I build-up to these two techniques through a series of equations, presented next.

Consider how a HMM may be applied to an observation sequence to compute an acceptance probability. The acceptance probability α1, is defined as follows: $\alpha 1 = P(O_1, O_2, O_3, \ldots O_R | \lambda)$. The acceptance probability of the next overlapping sequence of length R is $\alpha_2 = P(O_2, O_3, O_4, \ldots O_R | \lambda)$. Now, consider comparing the two acceptance probabilities:

$$\Delta\alpha = \alpha_2 - \alpha_1 \tag{7}$$

$$\%\Delta\alpha \overset{\text{def}}{=} \Delta\alpha/\alpha_1 \tag{8}$$

I can also consider non-overlapping sequences (aka data windows). Let the probability $\beta_2 = P(O_{R+1}, O_{R+2}, \ldots O_{R*2} | \lambda)$.

$$\Delta\beta = \beta_2 - \alpha_1 \tag{9}$$

$$\%\Delta\beta \overset{\text{def}}{=} \Delta\beta/\ \alpha_1 \tag{10}$$

The preceding equation (10) is technique 2, of comparing acceptance probabilities of consecutive data windows. The technique applies as follows:

1. Observation sequence O_1 is observed, $\alpha_1 = P(O_1 | \lambda_1)$
2. Observation sequence O_2 is observed, $\beta 2 = P(O_2 | \lambda_1)$
3. $\%\Delta\beta$ is computed
4. If $\%\Delta\beta \geq \text{Threshold}\beta$, then a transition is recognized

Technique 1, of comparing consecutive HMMs, is computed as follows:

$\Delta\lambda = \lambda_2 - \lambda_1$ **(111)**

To find the distance between two HMMs, I apply the widely applied Kullback-Leibler algorithm [64]. What's noteworthy here is that there are two different HMMs generated from two consecutive observation sequences. To see why, consider how each HMM is generated from an observation sequence.

Given an observation sequence, O, the emission probabilities matrix, B, is generated from the observed distribution of the symbols from V found in O. In my application, this is simply done by counting the email event, V, to determine their distribution in O and then adjusting the probabilities according to the state transition probability matrix A. Thus, each observation sequence, O_i, is modeled by an HMM $\lambda_i = (A, B, \pi)$, where A and π are given, and B models the probabilities of O_i. Each HMM is improved through optimization, which updates A, B, and π for a better fit with O_i.

A HMM λ can be optimized to maximize the probability of an observation sequence, O. Let $\bar{\lambda}$= maximize $P(O|\lambda)$. Baum-Welch is widely applied algorithm to maximize $P(O|\lambda)$, which I use. Thus, each observed sequence results in an optimized model: $\bar{\lambda}_1 = \max P(O_1|\lambda_1)$, $\bar{\lambda}_2 = \max P(O_2|\lambda_2)$, …$\bar{\lambda}_n = \max P(O_n|\lambda_n)$.

The technique 1, of comparing consecutive HMMs, applies as follows:

1. Observation sequence O_1 is observed, $\bar{\lambda}_1 = \max P(O_1|\lambda_1)$
2. Observation sequence O_2 is observed, $\bar{\lambda}_2 = \max P(O_2|\lambda_2)$
3. $\Delta\lambda$ is computed
4. If $\Delta\lambda \geq$ Threshold λ, then a transition is recognized

Technique 1 directly compares two HMMs, each generated from observation sequences; the consecutive distributions of O_1 and O_2 are compared via their representative HMMs, $\bar{\lambda}_1$= and $\bar{\lambda}_2$. Technique 2 compares the acceptance probabilities of the observation sequences using the first HMM. The two techniques produce similar results—when two observation sequences are significantly different, both $\%\Delta\beta$ and $\Delta\lambda$ will be above their thresholds.

3.5.5. The Markov Stream-Mining Approach

Having described the use of HMMs in modeling event streams and identifying transitions, I next show how I apply the approach to analysis of TAL email client's, like Don from the Introduction section.

TAL Data Stream

As mentioned before, the TAL email client provides an automated custom logger. To obtain real-time data access, a log file can be monitored. Here is a simplified entry from the log.

```
09:48:41 NewMailEvent [id=765406159;in-reply-to=311149530;chars=770;words=179;sentences=16]
```

This logged event specifies the time, the program event, and its associated arguments. The example logs the arrival of a new email that is in reply to previous e-mail; the identity of the sender and receiver and characteristics of the e-mail message, such as its length, are also included. The significant event types are: read email, compose email, delete email, and new (arriving) email. A database view provides a continuously updated stream. The dataset for one client, Don, included 3,695,086 records occupying 737 MB in Microsoft SQL Server 2005. These data are assumed for the remaining discussion.

Markov Model Parameters

User behavior depends on the state they are in, according to the reference theories of cognitive rehabilitation and learning (Section 2). In particular, the probabilities of their actions differ with their state. Because some states of theory are contemplative, non-action-taking states, I only consider two user states: S = {typical, atypical}. These state are consistent with the reference theories (Section 2) and allow me to focus on the monitoring needs of the post-clinical team, i.e., identifying when learning transitions occur. The states effectively partition event sequence probabilities into two sets: typical and atypical.

I selected the observation symbols from the email event types. Although there are a variety of types, I selected V = {compose, read}. These are the focus of the email learning. For example, in the simplified system, there is no reply event. Instead, read message from buddy X, followed by compose message to buddy X is in effect a reply. Thus, routinized sequences of read and compose indicate typical behavior, while more randomized sequences suggest atypical behavior.

For the initial state probability vector, π, I chose P(typical)=0.9, P(atypical)=0.1. I derive these numbers from my interpretation of the reference theories (Section 2), which show that learning is comprised of mostly typical behaviors interleaved with atypical learning episodes of less routinized behavior. Optimization updates π and the emission probabilities matrix, B, to fit the observed sequence.

The emission probabilities matrix B, is the final HMM parameter. The emission probabilities are generated from the observed distribution of the symbols in the observation sequence. For example, an emission probabilities matrix B =

	Read	Compose
Typical	0.35	0.65
Atypical	0.65	0.35

These probabilities are obtained by counting the email event types, V, to determine their distribution in O and then adjusting the probabilities according to the state transition probability matrix A.

In summary, λ = (A, B, π) is parameterized as follows:

- The states in the model, S = {typical, atypical}, the two learning states of the client
- The observation symbols V = {compose, read}, which are events from the email system
- The state transition probability matrix A =

	Typical	Atypical
Typical	0.9	0.1
Atypical	0.1	0.9

- The initial state probability vector π = [P(typical)=0.9,P(atypical)=0.1]
- The emission probabilities matrix B is generated from each window of event data.

HMM Support in KNIME

As mentioned earlier I am using KNIME for all of the data stream mining processes in this research. KNIME comes with several built in nodes for data mining such as decision trees, clustering, neural networks etc. KNIME does not have a node supporting HMM but provides the ability to the user of creating a new node and plugging that into the KNIME frame work. The user can make any customization needed for the node as while as he or she adheres to the KNIME node plugin specification. Consequently I created several new nodes in KNIME that supports HMM. The two ones that used extensively in this research where HMMs supporting up to three states and thirty observation.

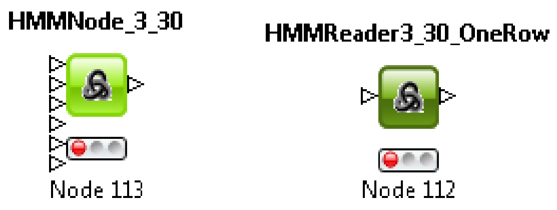


Figure 19 *HMM nodes for three states and thirty observations*

Figure 19 are the nodes been created in KNIME. The HMMNode_3_30 has six input ports and used for initializing the HMM creation are for 1) Number of States 2) Number of Observations 3) Initial State Probabilities 4) State Transition Probability Matrix 5) Emission Probability Matrix 6) The batch of data coming in each iteration as sequence of observations. The HMM gets these initial values and build an initial HMM object. The open source library in Java that utilizing forward-backward algorithm (Baum Welch) being used for creating the HMM object [92] and it uses several iteration to optimize the model. The optimization uses distance comparison between the newly created model with previous one (Kullback–Leibler divergence) and if the difference matches the predefined threshold then iteration stops. It is important to note that as the optimization uses specific form of expectation maximization to find a local optimum thus initial values for the input of the node is important to create the first HMM as close as possible to boundaries of the absolute maximum. The output port of the node is optimized HMMs presented as rows of a table in textual format for each sequence of observations. The HMMReader3_30 is a node having one input and one output. The input port accepts a sequence of HMMs and the output port generates a table having one column and each data in this column is the difference of each HMM with the previous HMM in the sequence coming from input. Having created these two nodes and using and wiring proper built in nodes in KNIME I can have a work flow that can analyze and simulate a stream of generated emails from the TAL user. The work flow is shown in Figure 20.

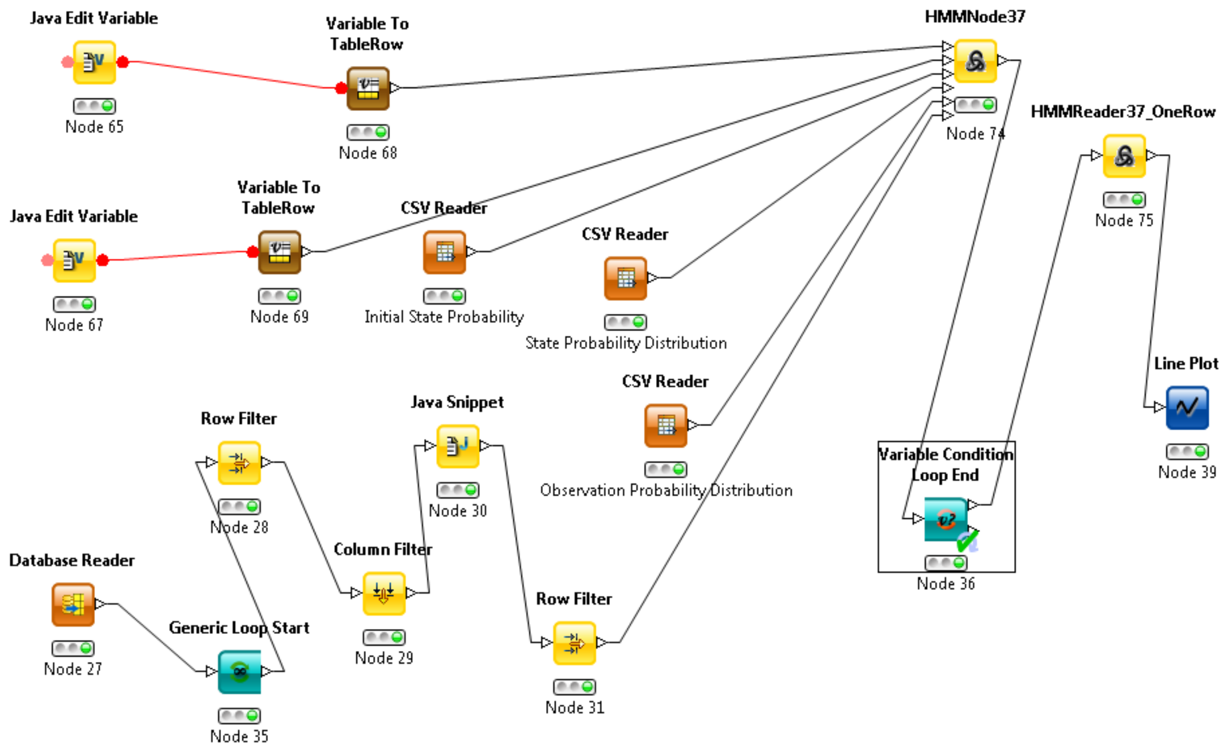


Figure 20 *The KNIME work flow for TAL user transition point discovery for three states and seven observations*

There is a loop node that simulates the whole static data in the data base as stream of data coming in as data window. The two created HMM nodes are generic and by integrating them into KNIME framework, I can create any workflow that requires HMM creation, reading and differencing. I used these two nodes in other work flows of my research. Next section I will explain the stream mining procedure for mentioned work flow more in details.

Streaming Email Markov Models

HMMs are created automatically for each data window, as they arrive from the data stream. After optimization, the two techniques from Section 3.5.4.2 are applied to identify transitions.

In stream mining, when the buffered stream data has more than window-size elements, data is transferred to the data window and processed. This process is repeated indefinitely. I selected a data window size of two-weeks, based on pre-testing and prior analysis of the TAL data [28] [72] [88] [89] [67] [29] [93]. This window size provides sufficient data for accurate modeling, a reasonable period to represent client behavior with the modeled states (i.e., time enough for typical or atypical behavior), and sufficient time to provide meaning feedback that the client is transitioning.

Within the two-week window, the data is divided into 4-hour segments. A data chunk aggregates like segments by weekday. Thus, the first Sunday segment (12 AM – 4 AM) is aggregated with the second Sunday. This provides (6 inter-day segments) x (7 days) = 42 data chunks. A HMM is created from each chunk. Thus, as each two-week data window is processed, 42 new HMMs are created. The reasoning behind the chunking is twofold: (1) it is the same as my prior modeling using decision trees, and thereby allows for direct comparison [72] [88], and (2) it assumes that behavior is mostly consistent by segment and day-of-week (i.e., clients do similar behaviors within a 4-hour period each day of week).

As a final step of processing, each HMM is improved through optimization using the Baum-Welch algorithm.

Transition Graphs

As presented in Section 1.3.3, transition identification detects significant changes in modeled events between consecutive windows of event data. Recall that a transition is a significant variation from one model of typical behavior to a new model of typical behavior (i.e., the new typical). This may be transitioning to a more knowledgeable state or relapsing to a prior state. I apply two HMM-based techniques: (1) comparing consecutive HMMs, and (2) comparing consecutive acceptance probabilities. Any change above a threshold is a transition, i.e., $\Delta\lambda \geq$ Thresholdλ or $\%\Delta\beta \geq$ Thresholdβ.

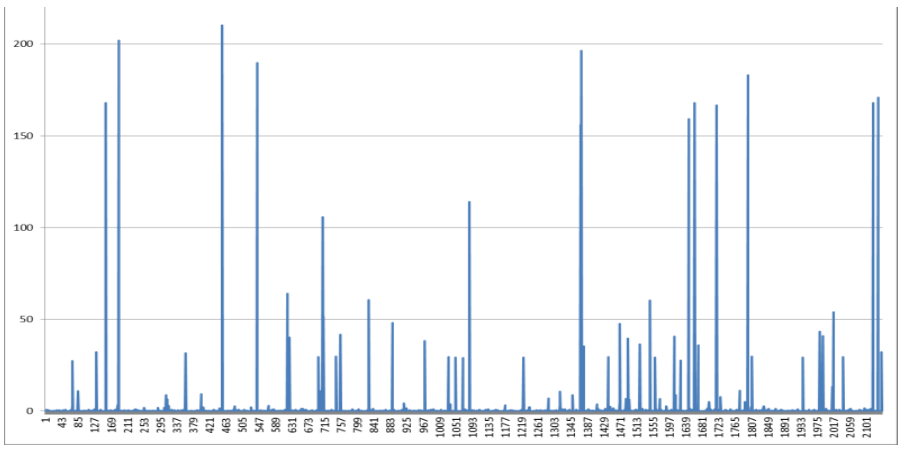


Figure 21 *HMM differences (by segment)*

Figure 21 graphs the result of using the Kullback-Leibler distance to compare HMMs (i.e., $\Delta\lambda$). The x-axis represents the comparison of the HMMs generated from the data chunks: (6 inter-day segments) x (7 days) = 42 data chunks, for 104 weeks (2 years) of two-week windows is 2,142 comparison points. Each point represents the comparison between two HMMs for the same segment from each bi-week; for example, Sunday segments 1 of weeks 1 and 2 compared with Sunday segments 1 of weeks 3 and 4.

The graph details of Figure 21 are not intended to be readable. Instead, notice the general trend. Many data points (the differences) are near 0—the average is 1.9. Nevertheless, there are 47 (of 2,142) points greater than 25. This suggests that the HMM differencing approach is good at discriminating unusual periods of sequential behaviors from the more common background.

Evaluation

Figure 22 also presents the HMM differencing approach (Δλ, technique 1). However, rather than showing the bi-weekly segment differences between HMMs for the 42 data chunks, the bi-weekly differences are totaled. The resulting line graph shows the aggregate bi-weekly differences. These differences allow me to consider how the two-week data windows are changing over time. Figure 22 also shows the differences of consecutive acceptance probabilities (%Δβ, technique 2), graphed as a dotted line. The graph scaled at 10 times (e.g., 10 x %Δβ) to aid visual comparison.

Notice that both %Δβ (technique 2) and Δλ (technique 1) have similarly shaped graphs; however, %Δβ is damped compared to Δλ, especially for small differences. This is because of the multiplication of the small probability values (< 10-2). In contrast, Δλ uses KullbackLeibler comparison, which is less sensitive to the small probability values. Thus, Δλ appears to be more sensitive, although its computation for large HMM is slightly more complex.

The bi-weekly HMM differences can be compared to prior bi-weekly difference analysis conducted using decision trees [28] [72] [88] [89] [67] [29] [93]. Figure 22 shows the bi-weekly decision tree differences as an area graph, for the same data. In comparing the HMM and decision tree differences, we see similar peaks, indicating that they both find differences in the same weeks (e.g. points 5, 13, 17, 26, 33, 37, 39, 43, 48). However, the HMMs and decision trees model the data differently.

The HMM models are the distribution of event sequences of length two and greater. The decision tree models the distribution of events regardless of their sequencing. Two different distributions of event sequences may contain the same distribution of unordered events;

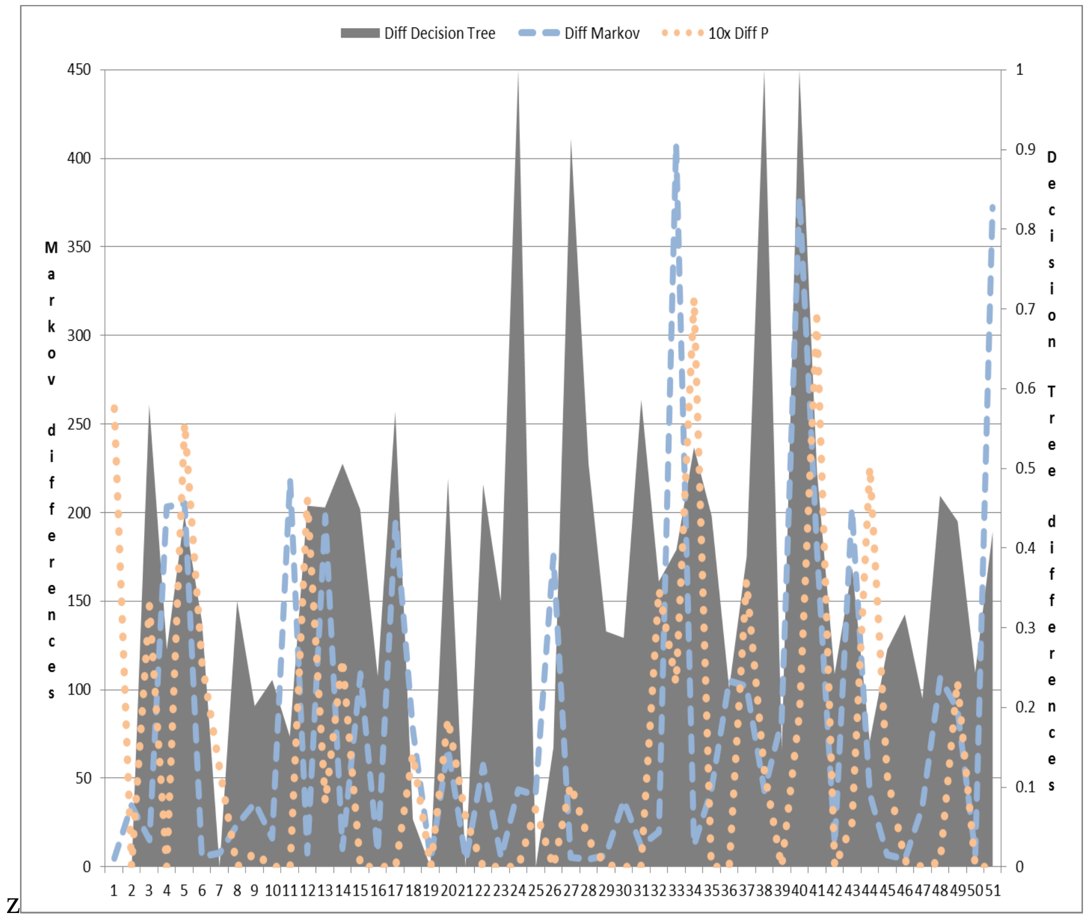


Figure 22 *Markov model differences (dashed line, scaled left) and probabilities differences (dotted line, 10x scaled right) over decision tree differences (area chart, scaled right)*

thus, a change in the HMM model does not necessary imply a change in the decision tree. A change in a decision tree model is likely to result in a change in the HMM. This arises simply from the fact that the decision tree model is indifferent to the sequencing that is central to the HMM.

Prior work has manually and automatically identified transitions in the client behavior for the data analyzed here [28] [72] [88] [89]. Figure 22 shows that the HMM transition analysis discovers transitions similar to the past studies, indicating that this approach is valid. This seems reasonable, given that we rely on well-known algorithms that identify sequence distributions and calculate their differences.

To confirm that there are indeed differences, I analyzed event sequences for each month of data associated with a significant peak in the HMM graph of 3.13. I applied Motif analysis. The Max Motif algorithm efficiently enumerates all maximal motifs in an input string [60]. Algorithm parameters consider the minimum occurrence threshold for pattern consideration, and the maximum of variables (aka wildcards) within the pattern. For example, B*AB**B is a pattern of length 7 with max wildcard length of 2. If the minimum occurrence threshold is 3, then this pattern will only be reported if it occurs 3 or more times in the input. The algorithm only reports on maximal motifs, which is a motif that is not properly contained by other motifs.

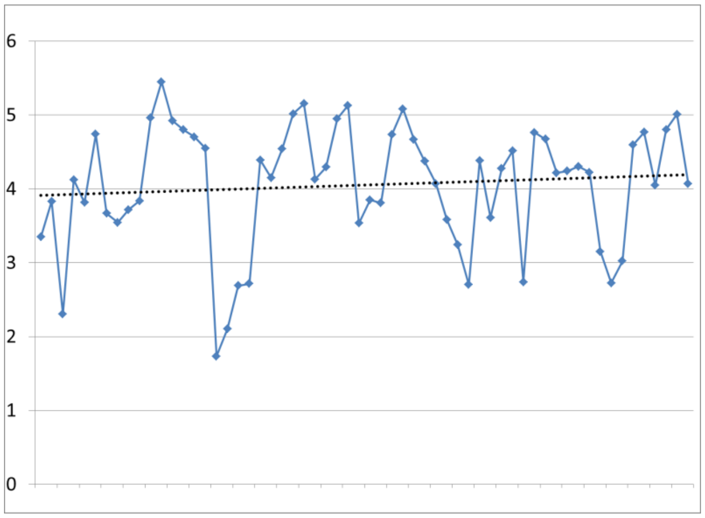


Figure 23 *Average non-variable pattern length*

To discover sequence patterns, I applied Max Motif to each window [89]. Motif analysis confirmed that identified transitions reveal changes in event sequences. For example, Figure 22 shows that point 26 has a transition. Point 26 compares the combined weeks 51 and 52 with the combined weeks 53 and 54. Figure 23 shows the read and write totals and the average sequence length per window around transition point 26.

All three factors decrease at point 26, confirming that something of interest occurred around point 26. Motif analysis combines well with HMM analysis. For example, from the Motif analysis, Figure 23 graphs non-variable pattern lengths per window, revealing a slightly increasing length [89]. Thus, it appears that the client, Don, is increasingly using sequential patterns, implying increased expertise in emailing.

3.5.6. Discussion

In this section, I showed how dynamically generated hidden Markov models (HMMs) characterize the distribution of sequence patterns in a software's user-interface event-stream. By differencing the resulting sequence of generated HMMs, this approach can identify transitions in software usage. This has been important for identifying behavioral transitions in clients with cognitive impairments as they learn to use their customized email system. When such transitions occur, caregivers provide assistance to ensure the client is not relapsing and encouragement to aid a progressing client.

I verified my approach by comparing the transitions it identified with those identified with other, independent techniques (e.g., decision tree differencing)—see Figure 24. Additionally, I applied the approach to four other clients, for a total of about 10 years of data. After reviewing the transitions, I find that it efficiently finds significant transitions from voluminous stream data. Members of the TAL team validated that the transitions were significant periods in each client's learning.

This stochastic approach to monitoring provides the following benefits:

- Discovery of sequence patterns and their probabilities of occurrence over streamed data windows
- Recognition of transitions to new distributions of patterns, which can be interpreted as new learning states
- Integration with diagnostic mining methods (e.g., MaxMotif), which characterize the exact nature of the transition, through presentation of the pattern differences

These monitoring features, when used in conjunction with compliance checking approaches, provide for a more comprehensive approach to runtime monitoring.

Next research section will address the issue of learning and learning durability. That is, what transition points are considered learning and does the client continue to apply the learned behavior, or is the learning lost? Additionally, the monitor should interpret its results in terms of user goal satisfaction. Addressing these issues will further simplify the monitoring activity so that caregivers can devote more time to caretaking and less to data analysis.

3.5.7. Experiment Conclusion

Software execution generates event sequences that should comply with its requirements. Hidden Markov models (HMMs) are good for characterizing the distribution of sequence patterns. Thus, HMMs are well suited to characterize software events. This is especially true for monitoring human activities where a threshold of variance can be

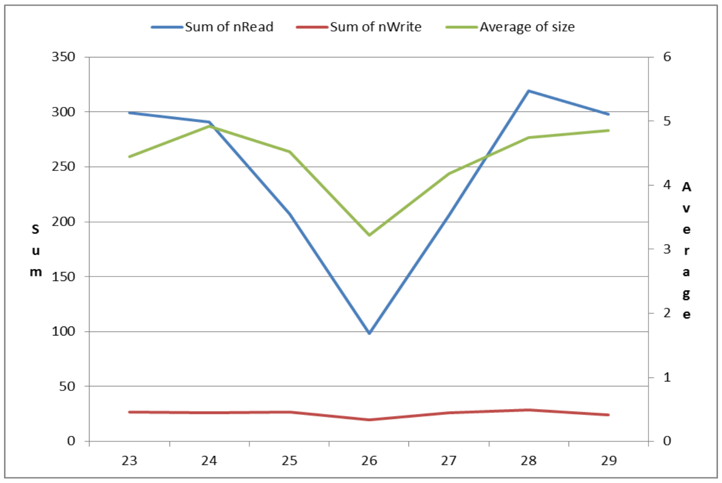


Figure 24 *Total read and write and average sequence length per window around transition point 26*

established to distinguish acceptable from worrisome behaviors; thus, it appears well suited to health-care monitoring. HMMs can be used to characterize software usage, look for unusual behaviors, and guide diagnosis of significant behavioral changes, as I demonstrated herein. Conventional monitors do runtime compliance checking of specified properties. Such monitors can increase their event coverage and analysis by analyzing statistical models, such as those represented in HMMs. Statistical models can also discover unanticipated legal ways in which users employ software. Thus, HMMs are a powerful technique to consider when designing a monitor.

3.6. Identifying Learning in TAL Users

3.6.1. Learning

The concept of learning and how we learn is a matter of research. However, some common definitions of learning are as follow:

- "Learning is an enduring change in behavior, or in the capacity to behave in a given fashion, which results from practice or other forms of experience." [94]
- "To gain knowledge, comprehension, or mastery of through experience or study." [95]
- "Learning is a process in which the learner attends to surrounding circumstances and is changed by exposure to them. [96]
- Learning is fundamentally about making and maintaining connections: biologically through neural networks; mentally among concepts, ideas and meanings; and experimentally through interaction between the mind and the environment [97]

Each of these definitions suggests we have to do something in order to learn. They suggest that learning occurs through experience, practice, and interaction with environment. Think about lyrics to your favorite song. How did you learn them? Did you interact with by playing them over and over again? Did you try to sing the lyrics every time you heard them on the radio or on TV? You were engaged in an active learning process and you learned them. Learning involves a permanent or at least a long-term change. Think about a learned song from long ago.. When suddenly you hear the song on the radio you sing the song just like you learned them yesterday. The learning of the lyrics represents a permanent change in behavior. When we learn we are changed. This change is enduring and might last for a long time. Sometimes we think we have learned but we really have not. You might have taken a quiz and got a good grade but when it comes to final exam you seem have a problem about the subject. This means you might have studied and memorized the subject but you really have not learned it.

The learning in this context can be categorized into two main categories:

- **Passive Learning**

  Passive learning is traditional way of learning. The teacher gives lectures to the students and provides them with the studying materials. Students listen and teacher shows them how to use the techniques and shows them solutions of some of the problems. During passive learning teacher does not much monitor the student's progress or does not comment on students work.

- **Active Learning**

  Active learning is experimental learning. The teacher asks the students to experiment with the procedure or the technique. The teacher monitors progress and tries to comment on each step. There are several other ways to engage students into active learning such as debate and discussions, showing videos, lab sessions, playing related games *etc*.

  Researches shows effectiveness of active learning. Richard Hake (1998) reviewed data from over 6000 physics students in 62 introductory physics courses and found that students in classes that utilized active learning and interactive engagement techniques achieved an average gain of 48% on a standard test of physics conceptual knowledge compared to a gain of 23% for students in traditional, lecture-based courses.

Instructors are shifting their methodologies to active learning. Students show more interest when learning is an active style learning. Figure 25 shows the cone of learning and illustrates the effectiveness of active learning compared to passive learning. As we can see from the figure, actions that are more experimental by the learner getting closer to active learning and consequently shows more effectiveness in learning the subject.

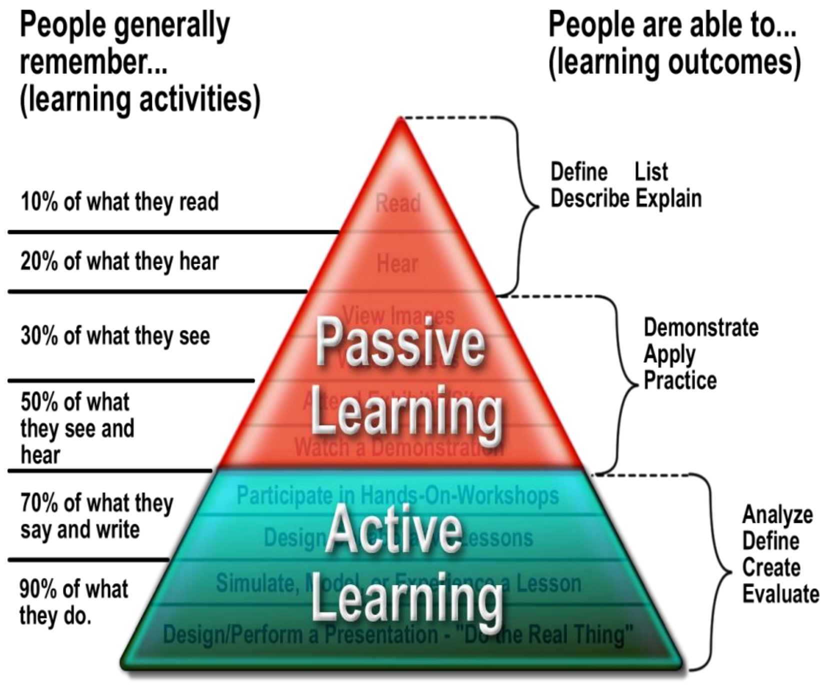

People generally remember... (learning activities)
People are able to... (learning outcomes)
10% of what they read
20% of what they hear
30% of what they see
50% of what they see and hear
70% of what they say and write
90% of what they do.
Read
Hear
View Images
Watch a Demonstration
Passive Learning
Participate in Hands-On-Workshops
Simulate, Model, or Experience a Lesson
Design/Perform a Presentation - "Do the Real Thing"
Active Learning
Define List Describe Explain
Demonstrate Apply Practice
Analyze Define Create Evaluate

Figure 25 *Cone of Learning*

In this research, we are studying cognitively impaired patients, thus their learning is a very critical issue. They are already in a situation that passive learning for them is extremely difficult task if not impossible so consequently helping these patients using any AT tool or application should be effectively utilizing active learning methodologies. Any tool should be designed in a way to provide experimental experience in every step of using it and making it an active learning tool. This, to an extent, ensures these patients stay interested using the application and gradually become integrated into society by using it.

3.6.2. Using hidden Markov models for Discovery the Learning Points

As mentioned in the first section, the Think and Link (TAL) project developed a specialized (closed) email system as part of a clinical treatment package. In TAL, the email software is personalized uniquely to each individual, as required by his or her treatment plan. Personalization is accomplished through evolution. User behaviors are continuously monitors to ensure that proper software adaptations are made to accommodate changing user needs. In contrast to most assistive technologies, TAL patients do not abandon the TAL email system. In fact, their emailing skills grow with usage, and in response, their unique email interfaces require continuous updates. In the commercial version of TAL, much of user monitoring is a manual process. This approach is not feasible for a large patient population. To address the scaling issue, TAL aims to automate much of monitoring in the research version of TAL. In particular, TAL's monitor must notify clinicians when unusual behavioral patterns against the background of typical behaviors. I discussed this already in previous sections using different stream data mining techniques to accomplish this automation. The two goals that I considered were simply 'read email' and 'compose and send email'. These two goals are directly supported by two observed events: ReadEmailEvent and ComposeMailEvent.

Computerized Adaptive Applications

I automate monitoring in TAL and help make it an active learning experience by finding the transition points in patient's usage behavior and keep customizing the application based on that.

One important thing to consider to make this customization more real-time, automated and having less care giver manual intervention is to identify if any learning happened in the transition points. The importance of this discovery is that if learning happened on any of those transition points then customization is needed in a way that gives the user more new-challenging features to the TAL application. On the contrary, if the transition point was not a learning point, the user struggled to learn the existing feature or typical usage of the application and consequently customization should be done in the direction making usage easier or less challenging. This process been applied to many CAT (Computerized Adaptive Testing) systems. These applications have pool of questions and questions are rated based on their difficulty. Every time examinee answers a question, the computer re-estimates examinee's ability based on all the previous answers and the difficulty of those items. The computer then selects the next question that examinee should have a 50% chance of answering correctly. This way, the next question should not be too easy or too hard. The computer's goal is to get as much information as possible about examinee true ability level and he should find each question challenging as each question is targeted to his ability. With each question answered, the computer's estimate of examinee ability becomes more precise.

The challenge I have for using adaptation to the customization of TAL is that this application is not an exam application and I am unable to adapt the customization based of previous correct/incorrect answers. That is why the measure for customization of the application is based on transition points and for more automated way of doing this I have to identify if those transition points are learning or non-learning. For the analogy to CAT application I can consider the series of correct answers are a match to learning point of TAL user and series of incorrect answers are a match to non-learning point of TAL user. Consequently when the adaptive exam application make the questions easier or more difficult depending to previous answers from the examinee, the TAL system should be customized in certain way to make it more suitable for the degree of challenge required for the user based on usage feedback that if he is learning or not learning.

Learning of TAL Users

As discussed I need to identify learning and non-learning for each transition points in the data being logged from TAL user. The fundamental question I have to answer is what concludes learning in TAL? In the section 3.6.1., I explained the learning defined as a process that causes a permanent or long lasting effect on the learner. The TAL user should communicate through this application with their friends. They should be able to receive, open, read, compose and delete emails during the day or night and at any hour. There are other features of TAL that are less important in my research such as add or delete friends etc. TAL users are especial case of email system users. Some have extreme difficulties to do typical email activities. For a typical user, usually the person has a routine for email activities. Just as an example, a full-time employee working in an IT department gets to his office around 9 AM, turns on his computer and start reading his emails. Then he decides to answer some of the emails that are important and leave the rest for later. After lunch around 1:00 PM he decides to compose some of the emails he left from the morning and possibly read some of other emails he received for today. Around 5 PM before he starts packing to leave for the day, he might read or compose some more emails. He usually checks email one more time around 11:00 PM before going to bed. This routine keeps repeating for the rest of the week. Of course, this routine might be different from person to person, title to title or organization to organization. Now imagine for the same employee routine changes for few days. He reads and composes emails every hour for about few days while at work. What we can conclude here? We can guess that there might be a production issue that changed his routine and making him to communicate continuously with other developers, application support, network team or database administrators. Or another guess can be that he was working on a project and the deadline getting close and that requires him be more in communication with other developers for integration testing and other related issues. Imagine another scenario; the same employee starts using email only for few minutes every day or not using email for few days at all. We might guess that employee is extremely busy with something that not requiring email communication at work or simply he is off and on vacation.

We can keep on going by different guesses for different email usage scenarios and we cannot be sure which one is happening in reality. Though what we can be sure that something especial happened in our imaginary employee professional or non-professional life. Now let us imagine that this email behavioral change we mentioned for this employee stays the same permanently or for a long period of time. Again we cannot guess what exactly happened but can be sure to an extent that something permanently or long lasting happened in employee's life. One guess can be that previous boss is gone and a new boss requires all employees be at work at 8:00 AM or sooner so email communication starting at that time. Or employee got promoted and his new title requires him to be in office sooner or make it more flexible to get to office a bit later. For TAL users similar analogy can be used but a key difference is that a permanent or long lasting change in email behavior of TAL user can be interpreted as learning. The user started using TAL with difficulty and had less motivation using it. The user then progressed and after a while learned how to use it more effectively. He can communicate with more friends and during different days and hours while before he was just ignoring receiving emails and not replying to them at all. This long lasting change of behavior for TAL user can be seen as learning to utilize the application better and more effectively. I discussed in previous sections that knowing a transition point to be learning is very important aspect of customizing the TAL application to make it more interesting and challenging for the user. On the other hand, if transition point is a non-learning point we need a customization that removes some barriers of using the application for the user and raising the excitement of the TAL usage.

Utilizing hidden Markov Model

I used hidden Markov models for identifying the learning points in the sample TAL user logging data I obtained for several months. A number of assumptions were made regarding the modeling and automation which analyzing of real data and the caregiver report to an extent endorsed validity of the assumptions. I used KNIME for my analysis (www.knime.org). As mentioned in section 3.7.5.3 there is no HMM support in KNIME and I created the nodes I needed for three states and thirty observations. I will be using the same nodes in my work flow for this research section. In addition to that, I created another KNIME node that will assist me in learning point discovery. Figure 26 showing this new node.

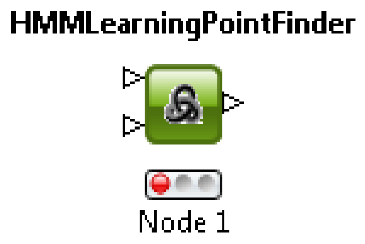


Figure 26 *The HHM learning point finder KNIME node*

The node has two input ports and one output port. The first input port tells the node that how many transition points (n) I am interested to examine if learning occurred. The second input port requires a sequence of HMMs that each HMM can be modeled for a sequence of observations in a specific time window. The node first compares each HMM with the previous one and finds the fluctuation pattern of HMMs. Then it finds top (n) highest peaks in this fluctuation which (n) comes from the first input port. After that for each transition point, comparison takes place between HMM before the point and after. The comparison is simply the distance between HMMs. Using weighted sum technique the HMMLearningPointFinder finds out that overall HMMs after the point are more similar to each other or to HMMs before the transition point. If they are more similar to each other that gets interpreted as learning and if more similar to previous HMMs then it is not a learning and transition point can be considered a sudden and short term happening or behavior. Each point will be represented with a number that depicts the strength of the learning. Figure 29 shows the KNIME work flow I created. More details about the technique and assumptions I used in my KNIME workflow are as follow: (Refer to section 3.5.2 for the HMM details)

a) I feed the application the 52 two-week data from TAL application. The day for each of these two-week are segmented into 6 sections (4 hour section). I make HMM of each day segment for each day week of each two-week. (For example, HMM for Sunday day segment between 12 PM – 4 PM for each two week)

b) I compare this HMM with next two week HMM and record the difference. I do the same for all day segments and week days. I add all the differences for all these HMMs and have an accumulative difference of two-week HMMs. Figure 27 shows the result of this process for the sample data.

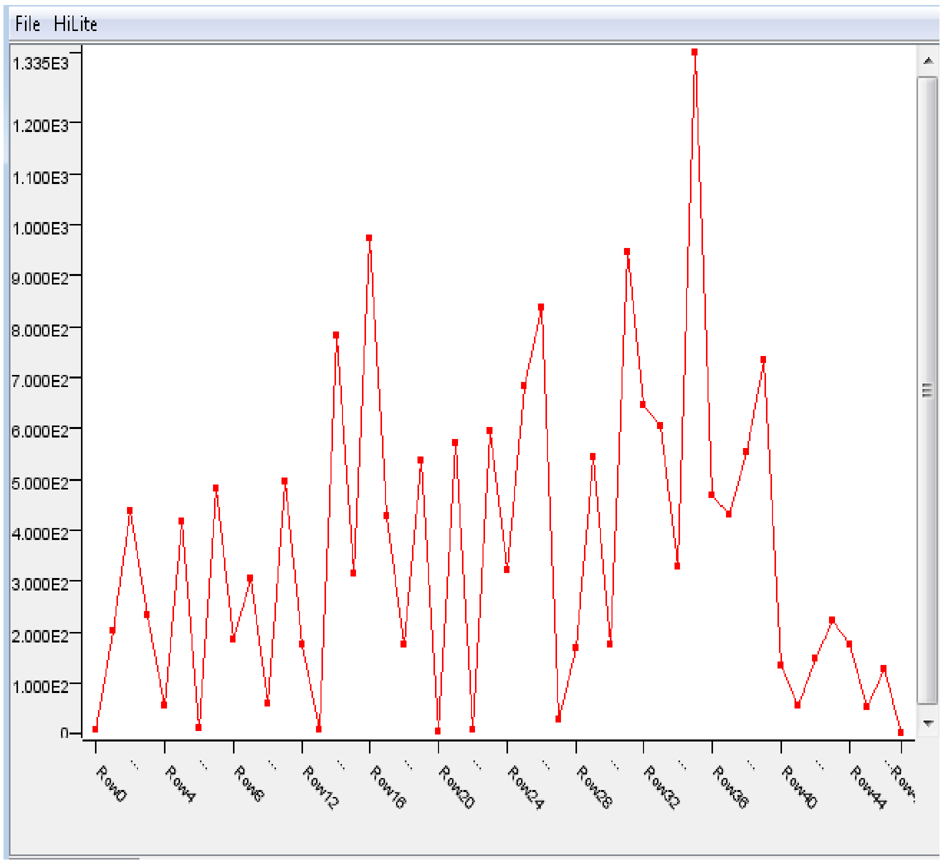


Figure 27 *The HHMs differences for 4 hour segments and two-week sliding window*

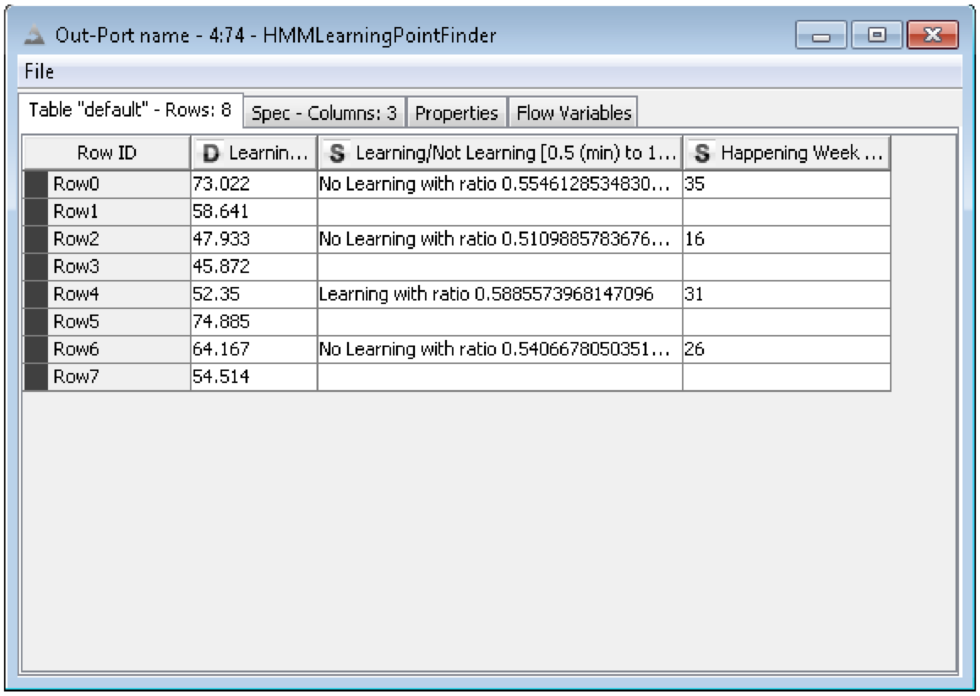
c) I provide the work flow the number of inflection points I am interested in. I requested workflow for *four inflection point discovery* and workflow searched the graph and found the top four critical ones. Obviously, the critical ones are the ones with highest HMM differences. In the provided graph (Figure 27) the work flow came up with 16, 26, 31 and 35 as top four critical inflection points. Obviously more than four inflection points can be requested to be discovered.

d) The workflow picks four two-week after and four two-week prior time range for each inflection point. Let us name these two time ranges A and B accordingly.

e) The workflow then for each inflection point takes first HMM from B and tries to find the distance of that HMM with every other HMM in A and B. Kullback-Leibler has been used for measuring the distance between two HMMs. If it finds the HMM with minimum distance in A range it gives one unit as a score to A otherwise it gives that unit of score to B. This gets repeated for every HMM in B.

f) At the end the score of A and B gets compared. If B has higher score means the inflection point was a learning point and the strength of learning can be measured as B / (A + B). Obviously this score will be between 0.5 (low learning) to 1.0 (high learning). Figure 28 shows the result from KNIME work flow. As we can see the only learning point happened was for third inflection point at 31st two-week window.

The reasoning behind this technique is that I try to see after infection point how much the sequence of user's behavior has changed for a reasonable period of time and actions don't have much similarity with the past. If that is the case, it means that the user following some activities that mostly happening after inflection point and they are new activities and keep repeating. While if most activities resemble prior inflection point activities then we might conclude that the inflection point was a sudden short-lived change in behavior, for example the user woke up very late in the morning and the whole emailing schedule for that day has changed but this not been the case for the following days.


Out-Port name - 4:74 - HMMLearningPointFinder

File

Table "default" - Rows: 8 | Spec - Columns: 3 | Properties | Flow Variables

Row ID	D Learnin...	S Learning/Not Learning [0.5 (min) to 1...	S Happening Week ...
Row0	73.022	No Learning with ratio 0.5546128534830...	35
Row1	58.641		
Row2	47.933	No Learning with ratio 0.5109885783676...	16
Row3	45.872		
Row4	52.35	Learning with ratio 0.5885573968147096	31
Row5	74.885		
Row6	64.167	No Learning with ratio 0.5406678050351...	26
Row7	54.514		

Figure 28 *The results from KNIME work flow discovering the learning points*

3.6.3. Conclusion

I briefly mentioned the several definitions of learning and all reflecting similar concept which is a permanent or long lasting effect on the learner. I discussed that there are different types of learning and categorized them based on their importance in the context of my research into active and passive learning. Based on statistics and some studying, most researches shown and believe that active learning is more effective methodology in learning than passive learning.

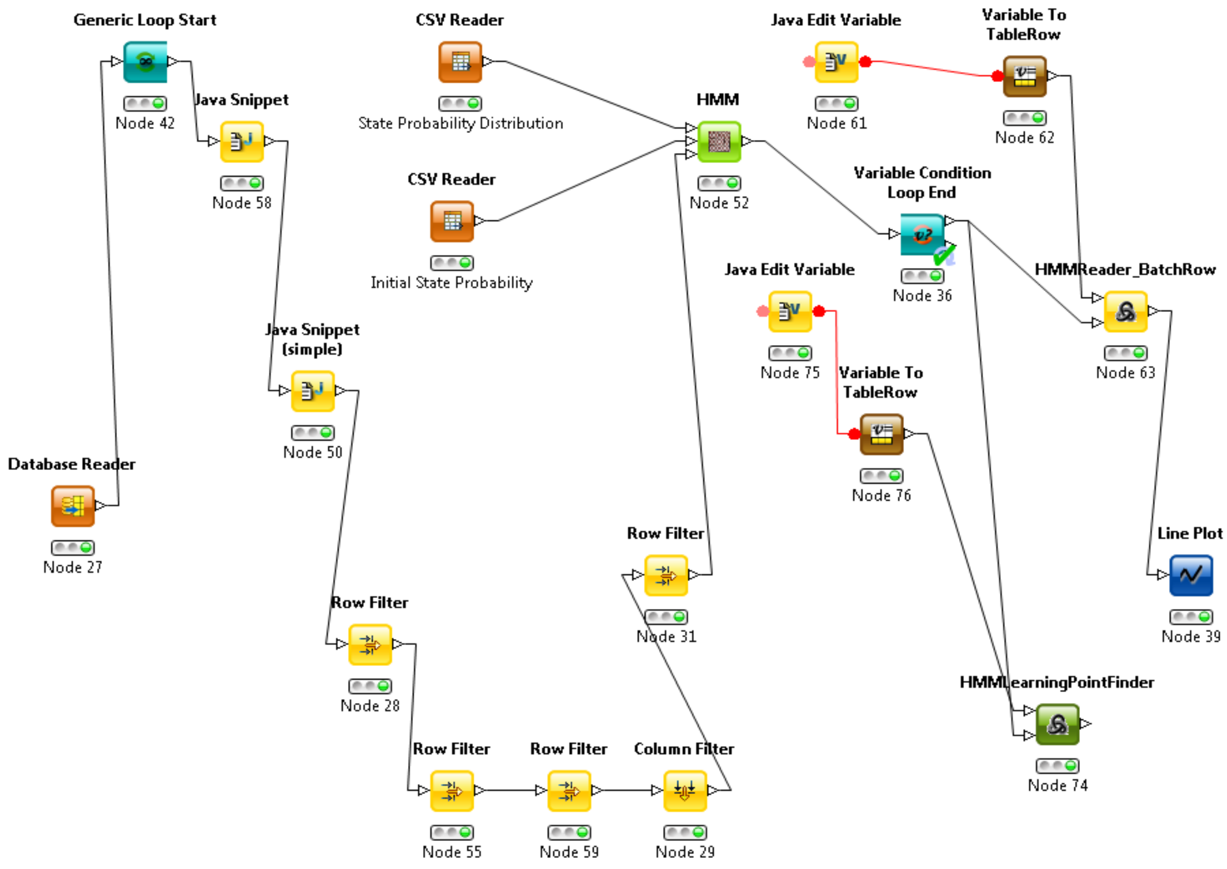


Figure 29 *The KNIME work flow for discovering the learning points*

One of the important criteria that should be considered to make learning an active process is to let the user have a complete experience of the application, challenge of using and learning of an application that being adjusted in different stages of user's learning process. Computer adaptive tests are applications that utilizing this concept. The application adjusts the difficulty and challenge of the questions based on history of the user correct answers to previous questions. Adaptation can also be used in applications that involve learning. The application needs to know at which stage it requires to make modification to the application and this will be the key question. The more precise these stages being identified the more effective the application will be and consequently the much better learning experience for the user. In my research I am dealing with TAL application that being used by cognitive impaired patients. These patients could easily abandon the system from frustration of being challenged more than their capability. Gradual and precise adaptive application is one of keys to success of making TAL extremely beneficial for this type of patients.

I made couple of assumption based on some facts and previous research experiences and concluded that stages or points I need to check the application for adaptation are the transition points in the user behavior. I considered starting from the most critical ones and the critical ones are the ones show the highest change of behavior. I assumed these points are critical as for several reasons the user might have changed behavior. One important reason is that the user based on previous experience learned to use that application more and better which in case of TAL means better and more communications with his buddies etc. If this is the case then the application requires adding more excitement and challenges to the user experience and increases the expectation from the user in a gradual way. Or during transition point user might have shown a short lived change of behavior and then restoring the previous behavior regarding the use of application. Now what constitutes that a transition point is learning or not? I assumed based on reasoning in section 3.6.2.2 that when a person changes behavior and that change last permanently or for long it means something has changed in that person's life and when this comes to the application and learning context the user of an application has learned something. I used hidden Markov models to see if sequences of actions after each transition point are more similar to after transition point or before. If more sequences of actions are similar to after transition points then it is learning otherwise it is not. I also rated the degree of learning using a weighted formula. The learning points are the points that TAL application should be adjusted considering that user has learned some stuff and adapt the application accordingly. The non-learning points consequently can be points that application does not need to change or might need to change in a way that user can use it more effectively and learn more. The results from this section were partially verified with the patient progress report and were satisfactory. Due to having limited data in my study, complete evaluation of several experiments required to have good judgments about using this technique, it validity and precision.

The future research can be focused in different areas of this initial studying as follow

- (a) Instead of using mentioned HMM simulation, different HMM modeling and simulation can be performed indicating if learning happened at transition points and results being compared for choosing most accurate one. Obviously modeling can vary from domain to domain and we can come up with suggestions that what kind of modeling suites specific domain.
- (b) The way adaptation should happen for different applications having different type of users can be a subject of more research.
- (c) Finding the rate of learning and if the user learned materials are vanishing by time
- (d) By getting access to TAL communication contents and performing text mining more learning metrics can be discovered about the user.

4. FRAUD DETECTION

In this section I am going to propose that in any domain that an entity generates some real-time data; the behavior of that entity can be analyzed using the previous techniques. The behavior transitional points, how critical they are and if transitional points are learning points (permanent or long lasting changes) can be identified. I applied this to real data from transaction processing engine of a leading company in prepaid industry for fraud detection and results were promising. My technique discovered fraud points that the company own fraud detection application fell short to identify. These developed techniques can open the door to many other domains and discovering some useful and interesting facts about their users. In case there is no real user, the simulation of a user can be done as I did in this section.

4.1. *Transactions processing at InComm*

Interactive Communications International (InComm) is a pioneer in prepaid technologies and solutions, InComm's distribution network of more than 75,000 retail locations is unparalleled. InComm was the first national point-of-service-activation (POSA) and distribution partner for all major wireless carriers -- and the only one with direct connection to the carriers. Though InComm provides many services and products, I am just concentrating in my research on their financial transactions. I briefly explain about a business scenario for one type of their financial transactions and then talk about their other services and product they offer to different customers. Though I focused on their financial transactions, I can apply my techniques on their other products and services as they all having similar business and transactional behavior.

The cards mentioned here are branded financial cards like visa or master card. The bank or financial institution based on the contract with InComm will create visa or master card accounts. These accounts will be married to proxy numbers by bank or the financial institution. Proxy numbers are just a number that can identify uniquely the account number. This number is not sensitive while account number is and can be safely used in transaction processing applications not considering the security protocols set for financial services. Each of these accounts has zero value and accounts are ready to be used if there is any amount of money in them. This file that has account numbers and proxy numbers will be transferred to InComm through a secure channel.

InComm based on this file will create a wrap gift card. The actual card which looks like a regular visa or master card will be inside a package that has a barcode and magnetic stripe on it.

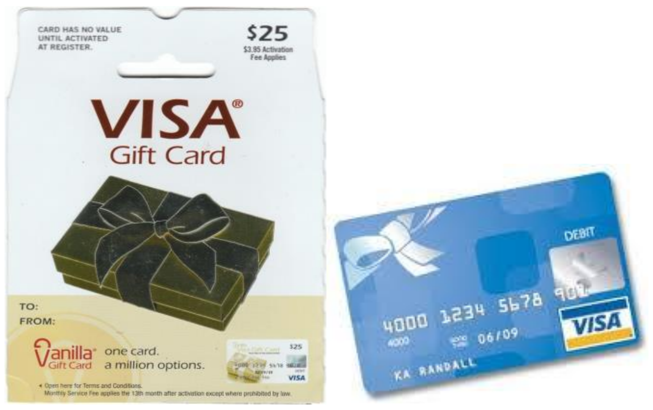

CARD HAS NO VALUE UNTIL ACTIVATED AT REGISTER.
$25
$3.95 Activation Fee Applies
VISA
Gift Card
TO:
FROM:
Vanilla Gift Card
one card. a million options.
DEBIT
4000 1234 5678 9010
06/09
KA RANDALL
VISA

Figure 30 *InComm Visa branded wrap gift card products*

The cards can be sold at any merchant having access to internet or dedicated data transfer lines. The customer takes the card to cashier and he/she will scan or swipe the package. Terminal is used by merchants to swipe the gift card magnetic stripe or they scan the bar code of the gift card. For many small merchants that do not have proper IT infrastructure the request transaction for loading certain amount of money for the swiped or scanned card directly hits the terminal server of InComm transaction processing center. For merchants that are large enough and have IT infrastructure their request transaction gets to merchant processing center (Some processes can happen on merchant side such as logging) then the request will come to InComm host-to-host (H2H) transaction processing center. At InComm transaction processing center, some business logic will be executed and proper formatted message including required data such as the proxy number and amount of transaction (card denomination) will be transferred to bank or financial institution. The bank will find the account number based on proxy number and add the amount requested in transaction to the account number. A successful response will be sent back to InComm and InComm will send back successful response to merchant cashier. This completes the sales. Now customer can open the package takes the inside card which looks like a regular visa or master card with no name on it and start using that any place accepts visa or master card. They can also call the bank or financial institution and setup a pin for using the card at the ATMs.

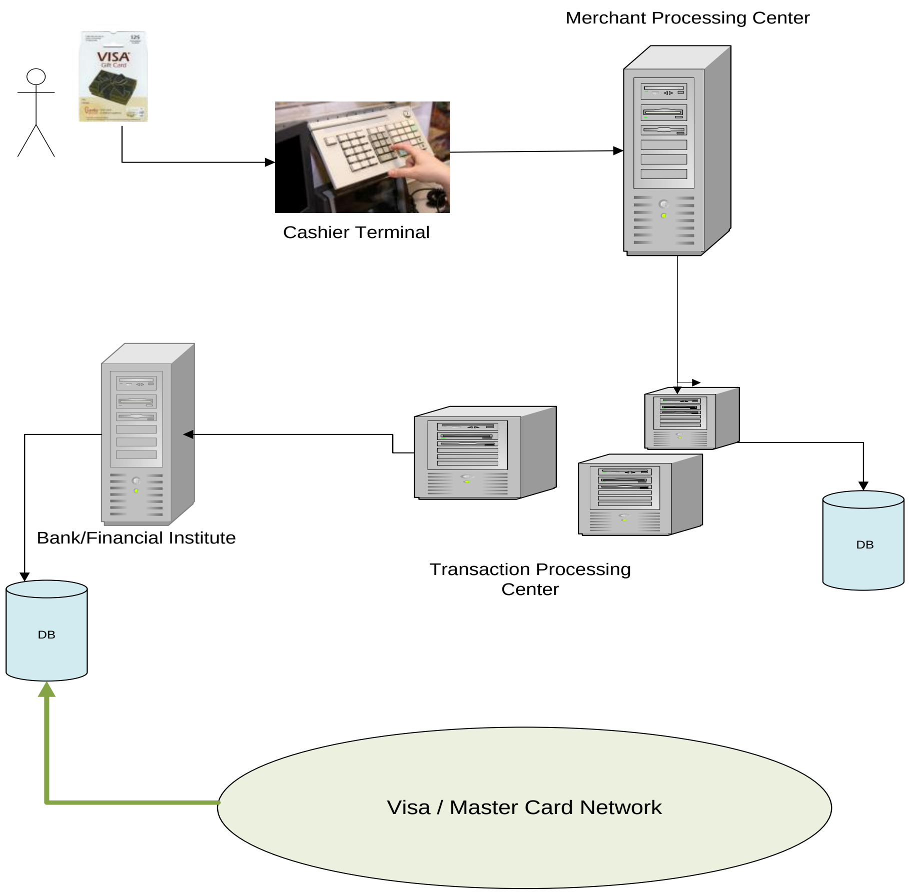

Merchant Processing Center
Cashier Terminal
Bank/Financial Institute
Transaction Processing Center
DB
DB
Visa / Master Card Network

Figure 31 *InComm gift card transaction processing flow*

InComm provide different services and products but most of them have similar concept in their transaction flow which means it is involving merchant, InComm and a third party player such as a bank. In general, InComm processing model connects Merchants, Consumer, Service Providers and Vendors with the goal of enabling a consumer to gain access to a product. These consumer products include:

- Wireless, which includes Boost and T-Mobile;
- Long Distance which includes Verizon
- GiftCard which includes private labeled gift cards such as Blockbuster;
- CreditCard which includes MyFastCardVisa and Mio;
- ThirdParty, which includes content providers such as Napster and iTunes.

The InComm transaction processing engine supports a number of business channels with its business partners, which include:

- Activation, where a product is Activated;
- Redemption, where the value is associated with a Consumer Account.
- Recharge, where a product is recharged
- Reporting, Reconcile, Billing & Product Setup channels, where a merchant, Vendor or 3rd Party is posted or requested information about the transactions processed by InComm;
- Payments, where payment is transferred

Connectivity by the business partners into the InComm system is enacted through a number of communication mechanisms which include:

- Kiosks,
- Interactive Voice Response and Call Centre Systems supplied by InComm & Vendors;
- Terminals at the Merchants;

- Point of Sale equipment at the Merchants;
- Access into the existing VISA network;
- Web and SMS interfaces.

There are several different categories of interface that are used within InComm that defines how a transaction is handled by the system. These include:

- Transferred Value, where Vendor reaches into the InComm system to perform the transaction;
- Real-Time & Near-Real Time , where InComm reaches to a Vendor/Service Provider to activate a product;
- Value Insertion, where InComm reaches to a service Provider to perform insertion of value into an account; and
- Pass Through, where InComm passes information to a 3rd party for activation

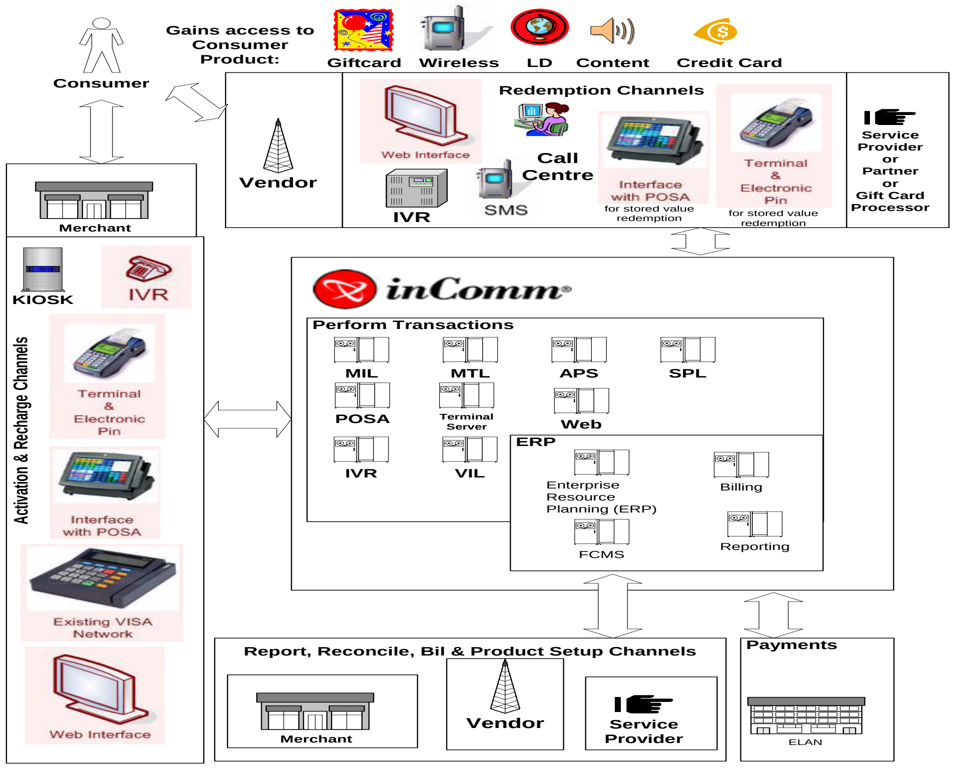

Consumer
Gains access to Consumer Product:
Giftcard
Wireless
LD
Content
Credit Card
Vendor
Redemption Channels
Web Interface
Call Centre
IVR
SMS
Interface with POSA for stored value redemption
Terminal & Electronic Pin for stored value redemption
Service Provider or Partner or Gift Card Processor
Merchant
KIOSK
IVR
Activation & Recharge Channels
Terminal & Electronic Pin
Interface with POSA
Existing VISA Network
Web Interface
inComm
Perform Transactions
MIL
MTL
APS
SPL
POSA
Terminal Server
Web
IVR
VIL
ERP
Enterprise Resource Planning (ERP)
Billing
FCMS
Reporting
Report, Reconcile, Bil & Product Setup Channels
Merchant
Vendor
Service Provider
Payments
ELAN

Figure 32 *InComm business context diagram*

Within the InComm solution there are various internal actors who as system interfaces enable the completion of the transaction between the various parties of the transaction. These are:

- MIL – Merchant Interface Layer
    - This manages the application of business rules to support communication from the Merchant to InComm. In some implementations, this layer also contains message translation, and can also support other sources of messages such as the VISA Network, Kiosks etc.
- MTL – Merchant Translation Layer

    - This manages the application of business rules to translate the merchant specific message structure into a generic InComm Structure.
- APS – Application Processing System
    - This manages the application of business rules for the transaction.
- SPL – Service Provider Layer
    - This manages the application of business rules and the interaction from InComm to the Service Provider. This layer may also interface with a Vendor system.
- VIL – Vendor Interface Layer
    - This manages the application of business rules from vendor into the InComm system. These typically include Vendor IVR or Web Site access to the systems.
- IVR – Interactive Voice Response System
    - This manages the application of business rules for IVR to access the InComm systems. This is typically used for redemption, and can also be used to support inter-transport to vendor hosted IVR systems.
- Web – Internet Web Site
    - This manages the application of business rules from Web site to access the Incomm systems. This is typically used for redemption.
- POSA – Point of Sale Activation
    - This manages the application of business rules to support communication with Point Of Sale (POS) equipment from the merchant into InComm.

- Terminal Server
    - This manages the application of business rules to support communication with Merchant Terminal equipment into InComm.

For all transactions the transaction will be logged at InComm Application Processing System (APS). Depends to the type of transaction, product or communication channel, APS uses different tables to log the activity. One of the tables that used heavily for this purpose by different transactions is ActivityLog table. The APS logs key information about each transaction which will be used for reconciliation with merchants or vendors. Each transaction will have two log records in this table. One is for the request and one is the response transaction. The activity log will have key information such as:

- TransactionId: Each transaction will be assigned a unique transaction Id.
- VAN16: Stands for Vendor Account Number and is the card number that is encoded in magnetic stripe or in the scanning bar code.
- Date & Time: Each transaction will have a date and time stamp.
- OpCode: Represents the action taken on the card such as activation or recharge etc.
- nNodeType: Indicates that where transaction originated from such as terminal or store or merchant or vendor etc.
- nRootNode: A number assigned to the transaction source. Like if the origination is terminal then this number will be terminal number.
- Amount: The amount of the transaction.
- SerialNumber: The card serial number.

- CardId: Every card has an Id which by using this id and joining other product related tables we can get information about the product being sold

There are many other tables being used by APS application but the AcitivityLog mentioned above is the major one that having real-time information about each transaction. The majority of other tables having static data regarding products, merchants, stores etc. which getting updated manually once a while per business need. There are few other tables similar to ActivityLog that are for some other products thus without loss of generality, applying any experiment or technique on ActivityLog can conclude the similar results on other tables. (Products)

4.2. InComm Fraud Detection Application

Individuals or groups of persons who make purchases over the internet or at store with the intent of cheating the Merchant are guilty of Fraud. Several factors such as faster purchases (sometimes over multiple locations in rapid succession), no paper trail, no visible contact and little risk of being caught (or successfully prosecuted) leave the merchant more vulnerable to fraudsters. According to Visa USA, fraud in traditional channels averages $.07 on every $100 in card transactions. Comparatively, card fraud through online channels is estimated to be between four to ten times higher. Fraud personas can fall into one of four types:

- **Friendly Fraudsters**: Legitimate cardholder’s friend or relative placing orders that has been stolen or borrowed for illicit purposes
- **Opportunistic Fraudsters**: Stumble upon valid payment info and commit fraud
- **Organized Fraud Rings**: Large, highly sophisticated groups often operating outside the US. Characteristic of changing methods and locations to thwart the latest fraud prevention methods.

- **Internal Fraudsters**: Employees of companies with secure cardholder data. They may be involved by giving external fraudsters access to valid payment data as well as the latest methods of foiling prevention techniques.

The inherent risk of fraud is a concern as InComm's electronic channels represent a significant growth and profit opportunity. A successful fraudulent attempt causes InComm to lose more than just the value of the stolen card. Cost incurred from online fraud place InComm (or its merchants) with the added liability of the transaction costs and charge backs. Excessive charge backs could result in further action being taken by the card associations such as being charged with higher transaction fees, having funds held in reserve, or even facing termination of service. The three prominent methods used to commit fraud against internet merchants are:

- **Stolen Cards**: Stolen and used before owner detection.
- **Identity Fraud**: Fraudsters assume the identity of the card holder.
- **Card Generators**: Fraudulent card numbers generated using software programs.

Being able to recognize high-risk transactions is crucial to identifying internet fraud. Some of the most common characteristics of fraudulent transactions include the following:

- Multiple purchases from same IP address on the same day.
- Orders from those using free email services, such as hotmail.com or juno.com.
- International orders, especially those that originate from high-fraud regions of the world. Currently, Nigeria, Indonesia, and Eastern Europe are producing a disproportional amount of online fraud in the US.
- Orders from first time buyers.
- Several of the same or repeated pattern of different item orders.

- Larger than normal orders.

While acting as an insurance against catastrophic fraud outbreaks, effective fraud prevention lowers the true cost of sales. Most of the fraud cases at InComm had a transactional footprint s which means the fraudulent transactions would show at InComm logging tables. Going throw several cases of fraud, InComm requested its IT department come up with fraud detection application. Transaction monitoring System (TMS) which went into production end of 2011 was officially InComm fraud detection application. TMS monitors sales activities for all retail locations for which the system is configured. The system monitors both sales volume (dollar amount of transactions) and transaction volume (number of transactions) for a merchant location in real time, as well as a daily sales transaction average.

The Fraud team sets up fraud monitoring rules in TMS that trigger alert notifications when configured thresholds, or trigger points, of sales or transaction volume, or a configured percentage of the daily sales average, for a product are reached for a location. The team has the ability to apply a rule to all stores for a merchant, or the team can choose specific stores to include and others to exclude from the rule. The team also chooses the products or product categories, or both, to which a rule applies. Rules can be modified or deleted, as required. The fraud team also has the ability to turn off, or disable, a fraud monitoring rule for a merchant. A rule can be configured to trigger a warning that alerts the Fraud team to a possible fraudulent situation at a merchant location. The team then can investigate and resolve the issue. A rule also can be configured to trigger a shutdown of transactions for a product or product category at a location. The fraud rules can be set up almost for every InComm product mention in section 3.7.1. All required information for TMS to trigger a rule coming from a table very similar in fields to APS ActivityLog mentioned previously. The table called tms_transaction_log which I talk about that more in next section.

Due to several cases of fraud and business need the Fraud Monitoring rules are as follow:

- **Sales Volume Monitoring:** A sales volume monitoring rule monitors the dollar amount of transactions for a product or product category and triggers alerts when a configured threshold is reached within a 24-hour period. Sales volume is calculated using the dollar (or other currency) amount of a location's transactions for a product or product category within a specified time period.
- **Transaction Volume Monitoring:** A transaction volume monitoring rule monitors the number of transactions for a product or product category and triggers alerts when a configured threshold is reached within a 24-hour period. Transaction volume is calculated using the transaction count of a location's transactions for a product or product category within a specified time period.
- **Daily Prepaid Transaction Sales Average Monitoring**: In TMS, a Daily Sales Average Rule is configured to trigger a warning alert when a location meets or exceeds 150% of its historical daily prepaid transaction sales average, which includes the combined sales of all products, within a 24-hour period. This percentage can be configured differently by merchant. Also, alerts can be configured for multiple percentages such as 175% and 200%.

For new locations, which do not have a historical daily sales average, the default amount trigger is $500. This amount can be configured differently by merchant. The fraud rules can be set up almost for every InComm product mention in section 3.7.1. All required information for

TMS to trigger a rule coming from a table very similar in fields to APS ActivityLog mentioned previously. The table called tms_transaction_log which I talk about that more in next section.

4.3. Fraud Detection by Identifying Behavioral Changes

As we can see TMS is a plain velocity monitoring system and lacks required character of an intelligent monitoring system. Though having TMS part of its transaction processing application was a success at InComm, it had an important shortcoming regarding discovering any fraud pattern in transactions. There are some past fraud cases at InComm that cannot be identified as fraud by TMS though by using previous techniques I already discussed, the fraud can be pin pointed precisely.

All the transaction for TMS being logged into tms_transaction_log and I use the same table to apply my modeling to. I follow the similar modeling and technique used in section 3.5.4. In the hidden Markov modeling, I defined a specific store of a merchant as an entity instead of the user of TAL application. I defined store can have three different states: typical, weekend and holidays. I analyzed the top selling product of that merchant and picked the top thirty of those products and removed the other products from tms_transaction_log. I defined the transition probability matrix as 3 by 30 dimension matrix that each value in this matrix represents the probability of selling one of the thirty products being in one of the three states. The sequence of different products being sold are the sequences that I used in my HMM modeling. And at the end I assigned equal probability to initial state probabilities of each state. Using Baum-Welch algorithm gives the confidence that any HMM we create will be optimized following the sequence of real observations which is the sequence of products being sold at that specific merchant location. I used similar technique mentioned in section 3.5.5. in order to see the change of behavior of the store regarding the product sales. Figure 33 shows my modeling using KNIME.

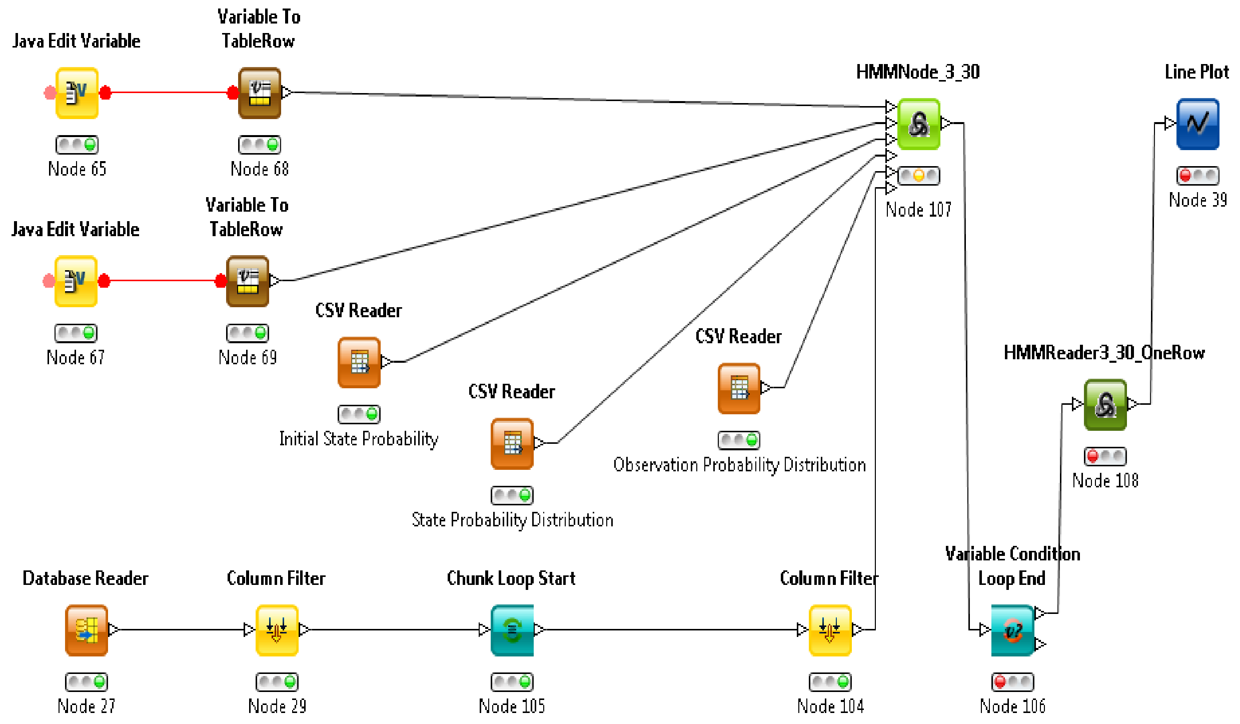


Figure 33 *InComm transaction hidden Markov modeling for three states and thirty observations*

I will mention few fraud cases and simulating them in the transactions. I will apply my modeling to those simulations afterward.

Fraud Case 1: An intruder got access to a terminal at the store and started swiping several similar financial cards using the terminal. This case can be simulated as a sudden sale of similar products in a short period of time. Therefore, I went and at a certain time in tms_transaction_logs injected a series of similar product being sold. The TMS cannot catch this fraud case as the series of the cards being sold might not push the total amount sold by the store above the threshold.

Alternatively, even if do so it might not be the same time this kind of fraud happening as threshold might reach alarming point end of the day. I ran my modeling against this simulated data for fifty transaction windows and it identified the time of this happening by an extreme change of the store behavior (maximum peak) in selling the products. Figure 34 shows the result.

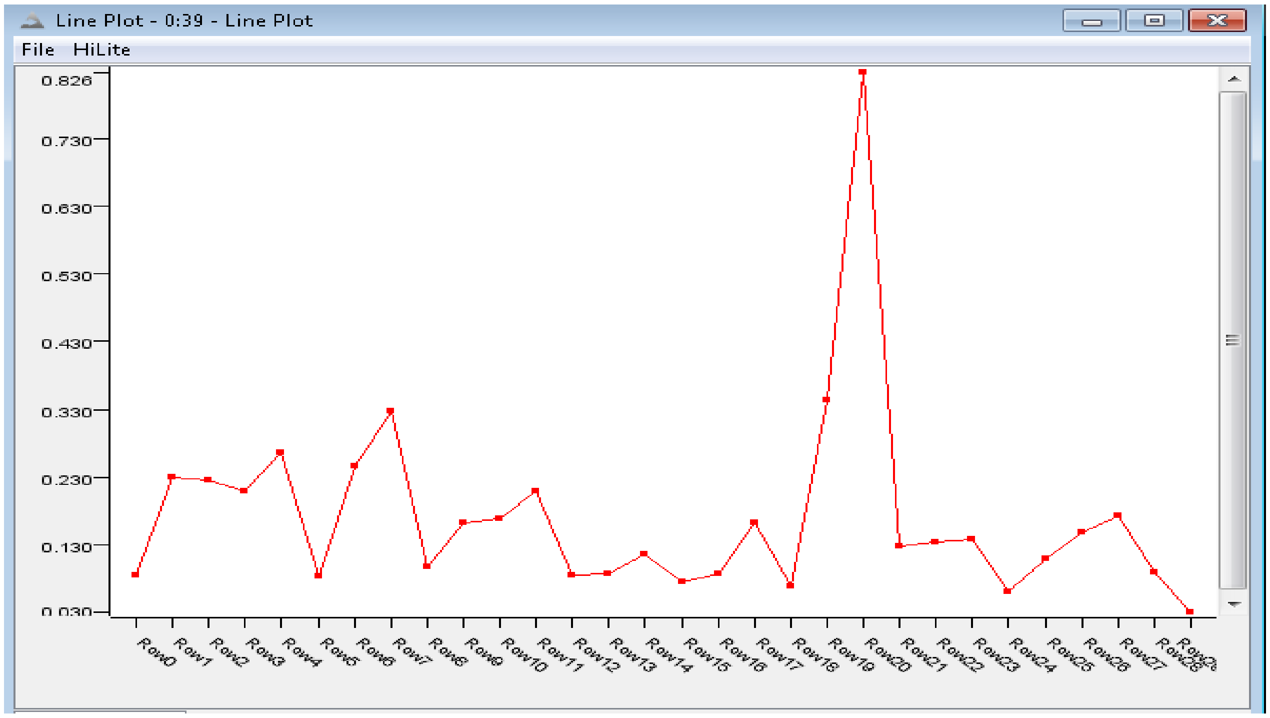


Figure 34 *The graph indicates that Fraud might have happened at the maximum peak (Around Row 19)*

Fraud Case 2: A group of intruders got to a store and each one picked a bundle of specific product and start swiping them against the terminals in store. In this case, few specific products will show as a sequence of transactions in tms_transaction_logs. I simulated this by getting four types of products and randomly generating a sequence of transactions using these four products at a specific time. I ran my modeling against this simulated data for fifty transaction windows and it identified the time of this happening by an extreme change of the store behavior (maximum peak) in selling the products. Figure 35 shows the result.

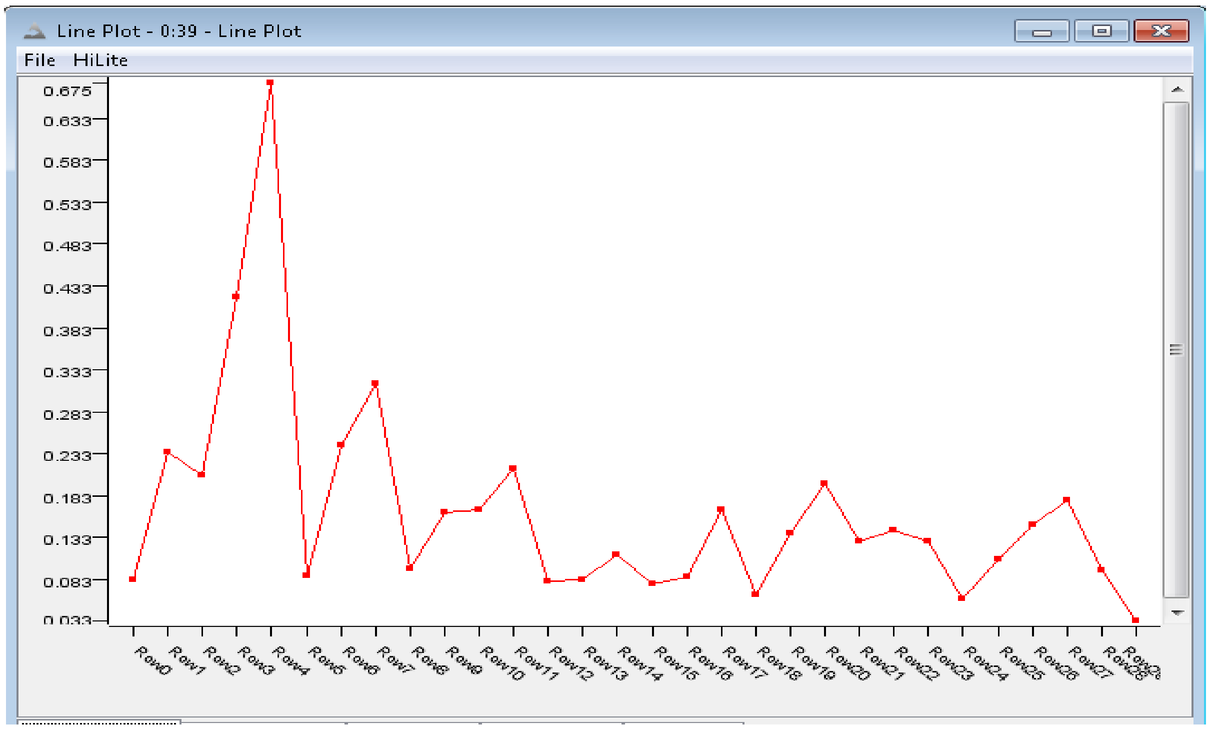


Figure 35 *The graph indicates that Fraud might have happened at the maximum peak (Around Row 3)*

Fraud Case 3: A fraudster acquired a serial range of several products and hacked into InComm network and start using a computer program to activate first serial range (first product) and then second serial range (second product) etc. I simulated this by picking five different products and start activating a series of first product followed by series of second product etc. I ran my modeling against this simulated data for fifty transaction windows and it identified the time of this happening by an extreme change of the store behavior (maximum peak) in selling the products. Figure 36 shows the result.

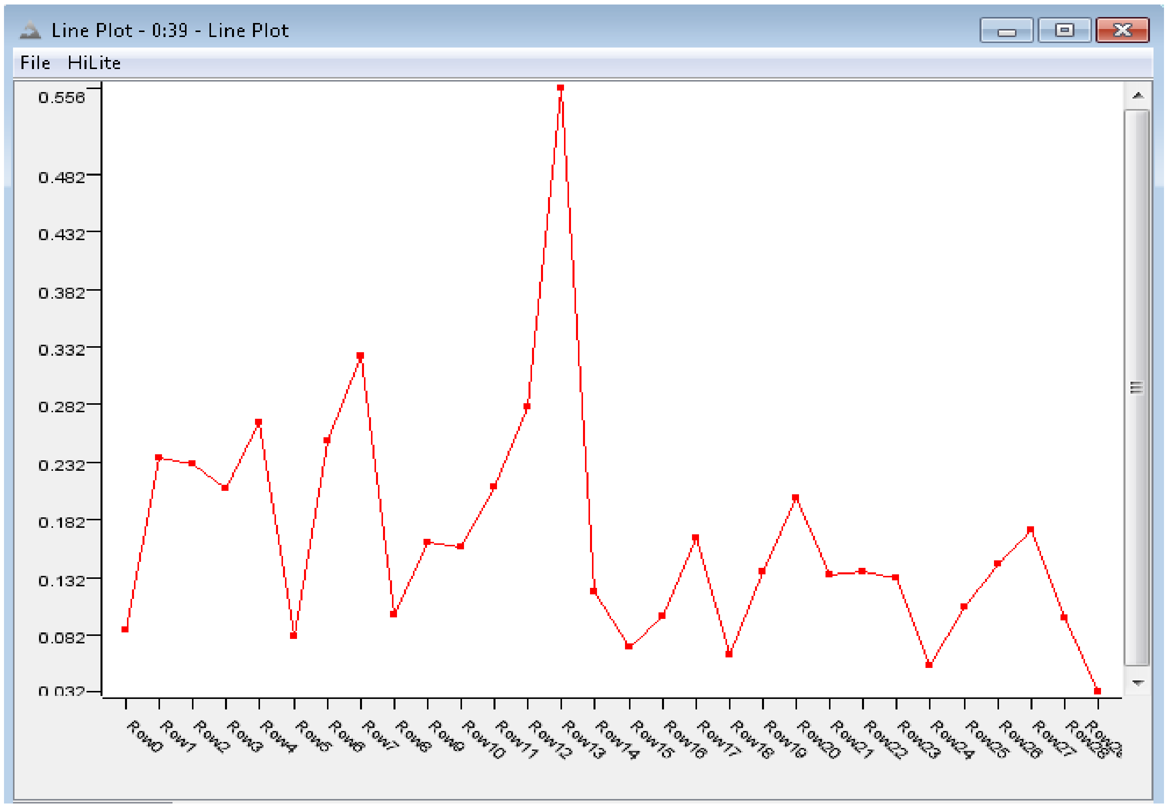


Figure 36 *The graph indicates that Fraud might have happened at the maximum peak (Around Row 12)*

4.4. Applying Fraud Detection Technique on Walmart Transactions at InComm

In the previous section, I imposed several fraud cases on some simulated transactional data. In order to examine the fraud detection technique and have better validation I applied the technique on real transaction at InComm and for specific merchant (Walmart). Without loss of generality the technique can be applied to any other merchant separately or at the same time. I gathered about 3 million transactions happened during early January 2013 from InComm transaction log archives. Walmart sells more than 100 different products at their stores. I picked the top selling store at those days and filtered the top thirty selling products. Consequently the transactions got reduced to around 400 thousand transactions. The same flow of Figure 32 being used for analyzing the fraud cases in these transactions and the similar fraud cases been mentioned in section 4.5 been injected at specific time of these transactions. Around 100, 120 and 150 transactions were picked as the size of sliding window for creating HMMs for different fraud cases. The size of window can be adjusted based on the extent we want to detect fraud and impact window size can be subject to another investigation and study. The smaller the window size the smaller number of transactions can be analyzed for behavioral changes that can at its extreme be interpreted as fraud. As we can see from the graph results of running the fraud detection work flow, the fraud incidents clearly can be observed as tremendous jump in the inconsistent behavior of the store. (Figure 37, Figure 38 and Figure 39) All the highest transitional points in the graphs been verified that actually are the points that fraud simulation happened. Another observation we can notice from the graphs is that using the smaller window in Figure 37 comparing to other figures indicates there are some other points that store might have change of behavior but not as extreme as fraud points. Those points can be anything from a short period of promotion offering, being out of stock for some products, a higher demand of especial product etc.

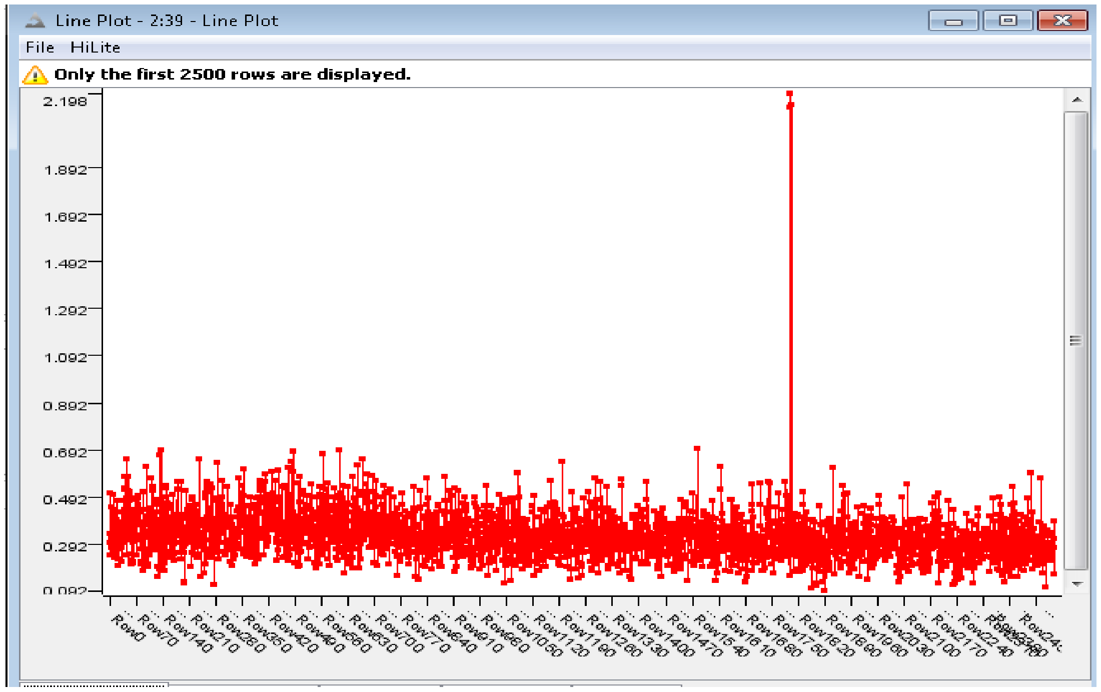


Figure 37 *Wal-Mart store graph transitional points for fraud case 1*

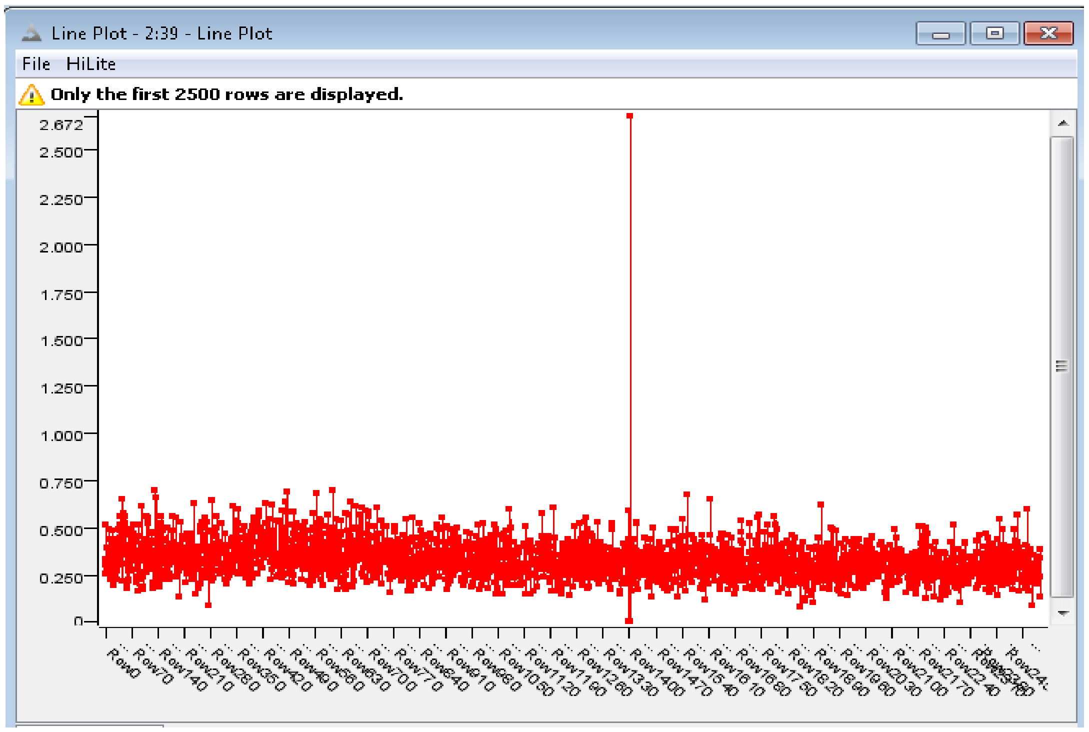


Figure 38 *Wal-Mart store graph transitional points for fraud case 2*

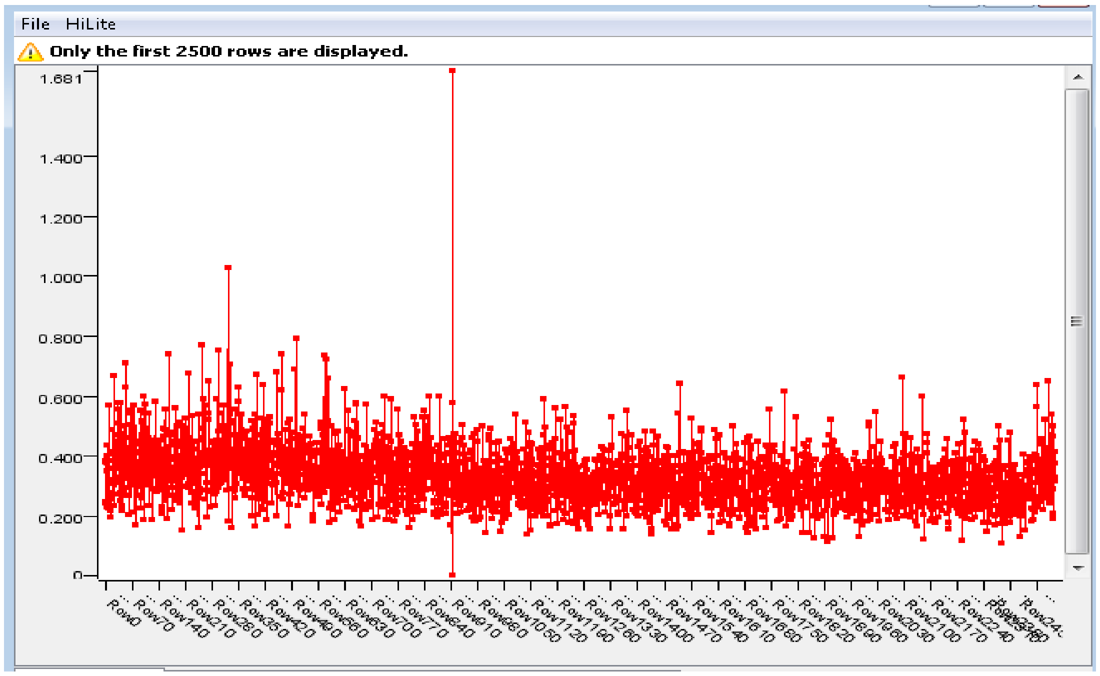


Figure 39 *Walmart store graph transitional points for fraud case 3*

4.5. Conclusion

I discussed that the hidden Markov modeling I developed in this research can be applied to several domains in order to explore the behavioral changes of the source that generates the underlying data. The previous section was dealing with TAL user and discussed the behavioral changes about his/her usage. In this section I picked InComm a leading company in prepaid business. I explained about their business model and variety of their products and channels of communications. InComm is a high volume transaction processing company and is dealing with growing fraud especially in its financial sector. They have a fraud monitoring system but as it is just a velocity check system I thought that this company is a great candidate for applying my modeling and see if it can assist them in better fraud monitoring. Consequently, I decided to apply my modeling to their transaction processing data. I simulated those fraud cases in the transaction logs and showed in my results that my modeling can catch the fraud at the happening time with a very good precision and in a real-time fashion. I discussed that existing company fraud monitoring system cannot catch these kinds of frauds that are pattern-based fraud and if it be able to do so, will not be a real-time discovery as TMS is a velocity fraud monitoring system. It means the amount of certain product at certain store should reach a certain threshold to trigger a fraud warning. In the simulated case, the threshold might never reach to defined threshold or it might reach the threshold not at exact time fraud has happened and too late for taking preventive actions. The other major advantage of my modeling is that it does not need a history of previous frauds and can be adjusted by time meanwhile the existing InComm fraud monitoring system rules should be adjusted once a while based on business needs or previous fraud history. A fraud monitoring system that includes both pattern discovery and velocity check should be much more beneficial to InComm in detecting variety of frauds.

My modeling was to an extent rudimentary, as I just wanted to show the proof of concept. Definitely modeling which includes more actors such as terminals of store, different locations of merchants and sequence of different actions on the same card might result into more and better discovery regarding fraud. My modeling can end up into many false positives as not every change in underlying transaction pattern is a fraud and this cannot be avoided. In addition to fraud discovery, my modeling can be used as an investigation of each store behavior for better marketing. Different stores for the same merchant can show different change of behavior during different times and that can give sales department better understanding about needs of the store regarding the products. This modeling can open an interesting door to sales, business, inventory management etc.

I hope that this section of research about InComm can help its IT department in designing a more intelligent fraud monitoring system to reduce the financial harm being caused by growing fraud in their growing business.

5. DISCUSSION

5.1. Research Objectives Revisited

In section 3.1, I discovered changes in model quality reveal changes in user behavior. My study illustrates this with the inflection points that Figure 11 summarizes. Note that there is a causality chain from the user's manipulation of the AT to the diagnosis of goal transitions. A user exercises the AT interface, such as reading and composing email. As mentioned in the same section, I discussed the results with the other TAL researchers. The goal transitions do seem to reflect persistent changes in behavior. Patient anonymity prevented me from directly correlating such real-world events, but it has been intimated that such events have occurred.

In section 3.4, analysis of sequence patterns supports clinicians in their efforts to link AT usage with cognitive theories. Pattern analysis can reveal user plans as variable sequences, even when a user makes occasional mistakes or pursues concurrent goals. The approach presented herein does not consider a statistical analysis of the discovered patterns. It would be helpful to know not just the occurrence counts for the patterns, but the statistical significance of the patterns within the data. It is possible to consider the pattern likelihood against the background of random sequences.

In section 3.5, I showed how dynamically generated hidden Markov models (HMMs) characterize the distribution of sequence patterns in a software's user-interface event-stream. By differencing the resulting sequence of generated HMMs, this approach can identify transitions in software usage. This has been important for identifying behavioral transitions in clients with cognitive impairments as they learn to use their customized email system. When such transitions occur, caregivers provide assistance to ensure the client is not relapsing and encouragement to aid a progressing client. I verified my approach by comparing the transitions it identified with those identified with other, independent techniques (e.g., decision tree differencing). Additionally, I applied the approach to four other clients, for a total of about 10 years of data. After reviewing the transitions, I find that it efficiently finds significant transitions from voluminous stream data. Members of the TAL team validated that the transitions were significant periods in each client's learning. These monitoring features, when used in conjunction with compliance checking approaches, provide for a more comprehensive approach to runtime monitoring

In section 3.6, I discussed the effectiveness of active learning comparing to passive learning. For TAL users, due to the fact that abandoning rate was very high; making the experience of using the application an active learning experience was a crucial thing. To know the learning points of the TAL user which means the points that we can assert that the user learned something new is a very important factor to adjust the application to maximize the usage of application and providing the users more challenges in this regard . On the contrary, if discovery was made that the inflection points in user behavior is not learning points and just an short-lived inconsistent behavior, ease of usage challenges of the application can be considered. Hidden Markov models being utilized for this goal and results were promising. By automation the whole idea into the TAL application, the hope is that application will be smart enough to provide the user the best it can regarding learning challenges and reduce the rate of abandoning the application.

In section 4, I explained that the technique used in section 3.7 can be applied to several other domains. Fraud detection domain was selected due to that fact that I am dealing with a company that trying to have a solid fraud detection system for years and had some real transactional data to share with us. I used previous hidden Markov model comparison and differencing technique. I first simulated some transactions and injected some fraud transactions in different sections and I was able to pin point the fraud point by running my application which utilizes the HMM technique. Then I acquired real transactional data from a well-known merchant and fraud transactions were injected into it. The application successfully detected the fraud scenarios. My study proved that this fraud detection application is superior in several aspects to existing threshold based fraud detection application of mentioned company. I suggested the combination of this technique and existing threshold based technique can be a more solid fraud detection system for similar transactional based companies.

5.2. Research Limitations

Generalizability of the approach is a concern and limitation in this research. Although initial results from this research were promising and could be verified, there were couple of research limitation that requires more study and research. More patient data needs to be used and same modeling techniques needs being performed on them. The transitional points need to be verified that matches the reality of patient behavior. The similar argument goes for the fraud detection section. Although I first simulated transaction data and then acquired real transactional data, still more data for different merchants and different shopping seasons required in order to make some general statements about validity of this technique and a robust fraud detection system.

This research conducted under a simplified user/entity behavior. A more complex behavior and modeling having several parameters required to verify that this technique can be applied in such environments too. In case of patient case study, having more actions and modeling and analyzing all those actions combined can be a better evaluation of patient behavior in reality though simplification is always an option. In case of learning, again I used a very simplified concept of learning despite the fact that many other parameters can determine the learning while using an application.

And finally, Hidden Markov models optimization always gives us the local optima. So as mentioned earlier finding a good initial values for HMM matrices are a crucial thing in finding the absolute optima. In my study a good guess was not a great challenge but further studying in different domains a good guess can be a crucial challenge to deal with and can have different outcomes if not being done properly.

5.3. Future Research

Modeling and technique discussed in this research can be applied to several other domains that required analysis of the behavioral changes of the source of generating data. InComm as a leading company in prepaid industry was chosen as a candidate to apply this modeling for enhancement of its fraud monitoring system. Its real transactions acquired and some fraudulent transactions got injected to its existing transactional data. The pattern discovery using my technique was successful to pin point the fraud points in exact time of happening.

The future researches can use this technique in several other domains and can be a subject for more research and development for many companies and institutions. Preliminary investigation makes this modeling suitable for market analysis and inventory management. The change of behavioral of a merchant or store can give a good feedback of inventory and market needs of that merchant. Similarly application of this technique and behavioral change of generating data source can be used as AI component for web usage regarding marketing and security, developing extremely adaptive applications for better learning, gaming industry, identifying network intrusion, stock price prediction, security in offices via security passes and many more areas. Hopefully there will be more researches applying these techniques in conjunction with other techniques and their results can be a better validation measure about some of the predictions and assumptions I made in this research.

6. SUMMARY & CONCLUSION

Date stream mining techniques can be used in tracking user behaviors as they attempt to achieve their goals. Quality metrics over stream-mined models identify potential changes in user goal attainment. When the quality of some data mined models varies significantly from nearby models—as defined by quality metrics—then the user's behavior is automatically flagged as a potentially significant behavioral change. The specific changes in user behavior are automatically characterized by differencing the data-mined decision-tree models. The decision tree modeling lacks considering the sequence of happenings consequently I used the same concept but utilizing sequence mining and hidden Markov models as they can reveal more information about different kind of changes in behavior. This has been implemented as a generic framework and applicable to many domains researching the change of user's or entity's behavior. The only requirement to make this framework applicable to other domain is the pre-processing of the input data and using a generic labeling technique.

To demonstrate the effectiveness of this framework, it has been applied to monitoring patient data in support of post-clinical care. The results of all three modeling were promising in discovering some behavioral changes of TAL users. These transitional points in user behavior can be analyzed and then proper modification be done to the application.

Later I focused on how I can make the adaptation of TAL application more automated based on these transitional points using this generic framework. This would give user better active learning experience and in a real-time fashion. Using hidden Markov models I tried to identify what transitional points in user behavioral changes are due to learning and which ones are just sudden short-lived changes. By identifying the learning and non-learning points I could make proper adaptation to TAL application and encourage the TAL users to get more engaged with the application and reducing the abandoning rate.

Finally I mentioned that the modeling and technique discussed can be applied to several other domains that required analysis of the behavioral changes of the source of generating data. InComm as a leading company in prepaid industry was chosen as a candidate to apply this modeling for enhancement of its fraud monitoring system. Its real transactions acquired and some fraudulent transactions got injected to its existing transactional data. The pattern discovery using my technique was successful to pin point the fraud points in exact time of happening.

APPENDICES

Appendix A: KNIME Implementations of HMM Nodes

HMMNode37

- The HMM node having three hidden states and seven observation

HMMNode_3_30

- The HMM node having three hidden states and thirty observation

HMMNode_3_50

- The HMM node having three hidden states and fifty observation

HMMReader37_OneRow

- The HMM node that reads data for three hidden states and seven observations one row at a time

HMMReader_3_30_OneRow

- The HMM node that reads data for three hidden states and thirty observations one row at a time

HMMReader_3_50_OneRow

- The HMM node that reads data for three hidden states and fifty observations one row at a time

HMMReader_BatchRow

- The HMM node that reads data for three hidden states and seven observations in batch

HMMLearningPointFinder

- The node that uses HMM to find out if the provided point as an input is a learning point

REFERENCES

[1] Tandon, G., and Chan, P. "Learning useful system call attributes for anomaly detection," 2005, pp. 405–411.

[2] Aggarwal, C., Han, J., Wang, J., and Yu, P. "On demand classification of data streams," ACM New York, NY, USA, 2004, pp. 503-508.

[3] Aggarwal, C.C., Watson, T.J., Ctr, R., Han, J., Wang, J., and Yu, P.S. "A Framework for Clustering Evolving Data Streams," 2003, pp. 81-92.

[4] R. Agrawal and R. Srikant, "Mining sequential patterns," in Data Engineering, 1995. Proceedings of the Eleventh International Conference on, 1995, pp. 3-14.

[5] Ajzen, I. "The theory of planned behavior," Organizational behavior and human decision processes (50:2) 1991, pp 179-211.

[6] Wang, H., Fan, W., Yu, P., and Han, J. "Mining concept-drifting data streams using ensemble classifiers," ACM New York, NY, USA, 2003, pp. 226-235.

[7] Alliance, F.C. "Incidence and Prevalence of the Major Causes of Brain Impairment," Family Caregiver Alliance, 2001.

[8] Sohlberg, M.M., and Mateer, C.A. Cognitive rehabilitation: An integrated neuropsychological approach Guilford Publication, New York, 2001.

[9] Services, U.S.D.o.H.a.H. "ADD Fact Sheet," Administration of Developmental Disabilities.

[10] Anderson-Lehman, R., Watson, H., Wixom, B., and Hoffer, J. "Continental Airlines flies high with real-time business intelligence," MIS Quarterly Executive (3:4) 2004, pp 163-

176.

[11] McColl, M., Carlson, P., Johnston, J., Minnes, P., Shue, K., Davies, D., and Karlovitz, T. "The definition of community intergration: perspectives of people with brain injuries," Brain Injury (12:1) 1998, pp 15-30.

[12] H. Arimura and T. Uno, "An efficient polynomial space and polynomial delay algorithm for enumeration of maximal motifs in a sequence," Journal of combinatorial optimization, vol. 13, pp. 243-262, 2007.

[13] Avison, D.E., Wood-Harper, A.T. Multiview: An Exploration in Information Systems Development Blackwell Scientific Publications, 1990.

[14] Azvine, B., Cui, Z., and Nauck, D. "Towards real-time business intelligence," BT Technology Journal (23:3) 2005, pp 214-225.

[15] L. E. Baum and J. Eagon, "An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology," Bull. Amer. Math. Soc, vol. 73, pp. 360-363, 1967.

[16] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," The annals of mathematical statistics, vol. 41, pp. 164-171, 1970.

[17] Sutcliffe, A., Fickas, S., and Sohlberg, M.K.M. "PC-RE: a method for personal and contextual requirements engineering with some experience," Requirements Engineering (11:3) 2006, pp 157-173.

[18] Sohlberg, M.M., Ehlhardt, L., Fickas, S., and Todis, B. "CORE: Comprehensive Overview of Requisite E-mail Skills," University of Oregon, Department of Computer and

Information Science, Eugene, OR.

[19] Beetz, M., Buss, M., and Wollherr, D. "Cognitive technical systems—what is the role of artificial intelligence?," KI 2007: Advances in Artificial Intelligence) 2007, pp 19-42.

[20] Bille, P. "A survey on tree edit distance and related problems," Theoretical computer science (337:1-3) 2005, pp 217-239.

[21] P. Bresciani, et al., "Tropos: An Agent-Oriented Software Development Methodology," Autonomous Agents and Multi-Agent Systems, vol. 8, pp. 203-236, 2004.

[22] D. J. Campbell, "Task complexity: A review and analysis," The Academy of Management Review, vol. 13, pp. 40-52, 1988.

[23] Chen, H. Medical informatics: knowledge management and data mining in biomedicine Springer Verlag, 2005.

[24] Chen, H., Fuller, S., Friedman, C., and Hersh, W. "Knowledge management, data mining, and text mining in medical informatics," Medical Informatics) 2005, pp 3-33.

[25] Sohlberg, M.M., and Mateer, C.A. Introduction to Cognitive Rehabilitation Guilford Press, 1989.

[26] Chung, L., and Subramanian, N. "Adaptable architecture generation for embedded systems," The Journal of Systems & Software (71:3) 2004, pp 271-295.

[27] Cook, J., Du, Z., Liu, C., and Wolf, A. "Discovering models of behavior for concurrent workflows," Computers in Industry (53:3) 2004, pp 297-319.

[28] Cook, J., and Wolf, A. "Discovering models of software processes from event-based data," ACM Transactions on Software Engineering and Methodology (TOSEM) (7:3) 1998, pp 215-249.

[29] Robinson, W.N., Akhlaghi, A., and Syed, A. "Diagnosing Stream-Mined Model Changes of Monitored Requirements for Cognitive Rehabilitation," Hawaii International Conference on Software Systems, IEEE, HI, USA, 2011.

[30] Fickas, S., Robinson, W., and Sohlberg, M. "The Role of Deferred Requirements: A Case Study," International Conference on Requirements Engineering (RE'05), IEEE, Paris, France, 2005.

[31] Sohlberg, M., Ehlhardt, L., Fickas, S., and Sutcliffe, A. "A pilot study exploring electronic (or e-mail) mail in users with acquired cognitive-linguistic impairments," Brain Injury (17:7) 2003a, pp 609-629.

[32]] Sohlberg, M.M., Ehlhardt, L.A., Fickas, S., and Sutcliffe, A. "A pilot study exploring electronic mail in users with acquired cognitive-linguistic impairments," Brain Injury (17:7) 2003b, pp 609-629.

[33] Sohlberg, M.M., Fickas, S., Ehlhardt, L., and Todis, B. "Case Study Report: The Longitudinal Effects of Accessible Email for Four Participants with Severe Cognitive Impairments.," Journal of Aphasiology, in press) 2005a.

[34] Sohlberg, M.M., Fickas, S., Ehlhardt, L., and Todis, B. "The longitudinal effects of accessible email for individuals with severe cognitive impairments.," Aphasiology (19:7) 2005b, pp 651-681.

[35] Sohlberg, M.M., Fickas, S., Hung, P., and Lemoncello, R. "Community Navigation Profiles for Six Individuals with Severe Cognitive Impairments, Poster session," The annual meeting of the National Academy of Neuropsychology, Seattle, WA., 2004.

[36] Sutcliffe, A., Fickas, S., and Sohlberg, M.M. "Personal and Contextual Requirements

Engineering," Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on) 2005, pp 19-30.

[37] Sutcliffe, A., Fickas, S., Sohlberg, M.M., and Ehlhardt, L.A. "Investigating the usability of assistive user interfaces," Interacting with Computers (15) 2003, pp 577-602.

[38] Todis, B., Sohlberg, M., and Hood, D. Insights into the needs for and barriers to using assistive technology in users with cognitive impairments, in press.

[39] Todis, B., Sohlberg, M.M., Hood, D., and Fickas, S. "Making electronic mail accessible: Perspectives of people with acquired cognitive impairments, caregivers and professionals," Brain Injury (19:6) 2005, pp 389-402.

[40] Pollack, M. "Intelligent assistive technology: the present and the future," User Modeling 2007) 2009, pp 5-6.

[41] Gaber, M., Zaslavsky, A., and Krishnaswamy, S. "Mining data streams: a review," ACM Sigmod Record (34:2) 2005, pp 18-26.

[42] Han, J., Chen, Y., Dong, G., Pei, J., Wah, B., Wang, J., and Cai, Y. "Stream Cube: An Architecture for Multi-Dimensional Analysis of Data Streams," Distributed and Parallel Databases (18) 2005, pp 173-197.

[43] D. Chudova and P. Smyth, "Pattern discovery in sequences under a markov assumption," 2002, pp. 153-162.

[44] Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G., Ng, A., Liu, B., and Yu, P. "Top 10 algorithms in data mining," Knowledge and Information Systems (14:1) 2008, pp 1-37.

[45] Hevner, A.R., March, S.T., Park, J., and Ram, S. "Design Science in Information Systems

Research," MIS Quarterly (28:1), Mar 2004b, p 75.

[46] Galin, D. Software Quality Assurance (from Theory to Implementation) Person, 2004.

[47] Jiawei, H., and Kambe, M. "Data mining: concept and technology," Beijing: Machine industry press, 2001.

[48] Lamsweerde, A.v. Requirements Engineering: From System Goals to UML Models to Software Specifications, Wiley, 2008.

[49] Robinson, W.N., and Fickas, S. "Talking Designs: A Case of Feedback for Design Evolution in Assistive Technology," in: Design Requirements Engineering: A Ten-Year Perspective, K. Lyytinen, P. Loucopoulos, J. Mylopoulos and W. Robinson (eds.), Springer-Ver, 2009, pp. 215-237.

[50] Hevner, A.R., March, S.T., Park, J., and Ram, S. "Design Science in Information Systems Research," MIS Quarterly (28:1) 2004a, pp 75-105.

[51] A. Dardenne, A. van Lamsweerde, and S. Fickas, "Goal-Directed Requirements Acquisition," Science of Computing Programming, vol. 20, pp. 3-50, 1993.

[52] Delgado, N., Gates, A.Q., and Roach, S. "A Taxonomy and Catalog of Runtime Software-Fault Monitoring Tools," IEEE Transactions on Software Engineering (30:12), December 2004, pp 859-872.

[53] C. C. DiClemente and J. O. Prochaska, "Self-change and therapy change of smoking behavior: A comparison of processes of change in cessation and maintenance," Addictive behaviors, vol. 7, pp. 133-142, 1982.

[54] Domingos, P., and Hulten, G. "Mining high-speed data streams," ACM New York, NY, USA, 2000, pp. 71-80.

[55] Fenstermacher, K.D., and Ginsburg, M. "A Lightweight Framework for Cross-Application User Monitoring," Computer (35:3), March 2002.

[56] Ferrer-Troyano, F., Aguilar-Ruiz, J., and Riquelme, J. "Discovering decision rules from numerical data streams," ACM New York, NY, USA, 2004, pp. 649-653.

[57] Fickas, S., and Feather, M.S. "Requirements Monitoring in Dynamic Environments," Proceedings of the 2nd International Symposium on Requirements Engineering, IEEE Computer Society Press, York, England, 1995, pp. 140-147.

[58] Fickas, S., and Helm, R. "Knowledge representation and reasoning in the design of composite systems," IEEE Transactions on Software Engineering (18:6), June 1992, pp 470-482..

[59] S. Flake, "Enhancing the Message Concept of the Object Constraint Language," in In Sixteenth International Conference on Software Engineering and Knowledge Engineering (SEKE 2004), Banff, Canada, 2004, pp. 161-166.

[60] G. D. Forney Jr, "The viterbi algorithm," Proceedings of the IEEE, vol. 61, pp. 268-278, 1973.

[61] W. N. Robinson, "A Roadmap for Comprehensive Requirements Monitoring," IEEE Computer, vol. 43, pp. 64-72, May 2010.

[62] W. N. Robinson and S. Purao, "Monitoring Service Systems from a Language-Action Perspective," IEEE Transactions on Services Computing, 2010.

[63] W. N. Robinson and S. Purao, "Specifying and Monitoring Interactions and Commitments in Open Business Processes," IEEE Software, vol. 26, pp. 72-79, July 2009.

[64] W. N. Robinson, "Seeking Quality through User-Goal Monitoring," IEEE Software, vol.

26, pp. 58-65, 2009.

[65] W. N. Robinson and S. Purao, "Development Support for Specifying and Monitoring Goals of Open Business Processes," in 4th International Workshop on Service-Oriented Computing Consequences for Engineering Requirements (SOCCER'08), Barcelona (Spain), 2008, pp. 53-64.

[66] W. N. Robinson, "Extended OCL for goal monitoring," Electronic Communications of the EASST, vol. 9, pp. 1-12, January 2008.

[67] Robinson, W.N., and Akhlaghi, A. "Monitoring Behavioral Transitions in Cognitive Rehabilitation with Multi-Model, Multi-Window Stream Mining," Hawaii International Conference on Software Systems, IEEE, Kauai, HI, USA, 2010.

[68] Gaaloul, W., Bhiri, S., and Godart, C. "Discovering workflow transactional behavior from event-based log," LECTURE NOTES IN COMPUTER SCIENCE) 2004, pp 3-18.

[69] Gama, J. Knowledge discovery from data streams Chapman & Hall/CRC, Boca Raton, 2010.

[70] J. Gama, Knowledge discovery from data streams. Boca Raton: Chapman & Hall/CRC, 2010.

[71] Gama, J., Ganguly, A., Omitaomu, O., Vatsavai, R., and Gaber, M. "Knowledge discovery from data streams," Intelligent Data Analysis (13:3) 2009, pp 403-404.

[72] R. Gwadera, M. J. Atallah, and W. Szpankowski, "Reliable detection of episodes in event sequences," Knowledge and Information Systems, vol. 7, pp. 415-437, 2005.

[73] Han, J., and Kamber, M. Data mining: concepts and techniques Morgan Kaufmann, 2006.

[74] Hulten, G., Spencer, L., and Domingos, P. "Mining time-changing data streams," ACM

New York, NY, USA, 2001, pp. 97-106.

[75] Jain, G., Cook, D.J., and Jakkula, V. "Monitoring Health by Detecting Drifts and Outliers for a Smart Environment Inhabitant," in: 4th International Conference on Smart Homes and Health Telematics, 2006.

[76] Kargupta, H., Bhargava, R., Liu, K., Powers, M., Blair, P., Bushra, S., Dull, J., Sarkar, K., Klein, M., and Vasa, M. "VEDAS: A Mobile and Distributed Data Stream Mining System for Real-Time Vehicle Monitoring," Proceedings of the SIAM, International Data Mining Conference, Orlando) 2004.

[77] E. Koechlin, et al., "The role of the anterior prefrontal cortex in human cognition," Development, vol. 1, pp. 157-167, 1991.

[78] Last, M. "Online classification of nonstationary data streams," Intelligent Data Analysis (6:2) 2002, pp 129-147.

[79] S. Laxman and P. S. Sastry, "A survey of temporal data mining," Sadhana, vol. 31, pp. 173-198, 2006.

[80] S. Laxman, P. S. Sastry, and K. P. Unnikrishnan, "Discovering Frequent Generalized Episodes When Events Persist for Different Durations," Knowledge and Data Engineering, IEEE Transactions on, vol. 19, pp. 1188-1201, 2007.

[81] LoPresti, E., Mihailidis, A., and Kirsch, N. "Assistive technology for cognitive rehabilitation: State of the art," Neuropsychological Rehabilitation (14:1-2) 2004, pp 5-39.

[82] Ludwig, H., Dan, A., and Kearney, R. "Crona: an architecture and library for creation and monitoring of WS-agreents " in: Proceedings of the 2nd international conference on Service oriented computing ACM Press, New York, NY, USA 2004 pp. 65-74.

[83] H. Mannila, H. Toivonen, and A. Inkeri Verkamo, "Discovery of frequent episodes in event sequences," Data Mining and Knowledge Discovery, vol. 1, pp. 259-289, 1997.

[84] F. Masseglia, P. Poncelet, and M. Teisseire, Successes and new directions in data mining: Information Science Publishing, 2008.

[85] Phua, C., Smith-Miles, K., Lee, V., and Gayler, R. "Adaptive spike detection for resilient data stream mining," Australian Computer Society, Inc. Darlinghurst, Australia, Australia, 2007, pp. 181-188.

[86] Robinson, W.N. "A requirements monitoring framework for enterprise systems," Requirements Engineering Journal (11:1) 2006, pp 17-41.

[87] W. N. Robinson, "Implementing Rule-based Monitors within a Framework for Continuous Requirements Monitoring, best paper nominee," in Hawaii International Conference On System Sciences (HICSS'05), Big Island, Hawaii, USA, 2005.

[88] W. V. J Prochska, "The Transtheoretical Model of Health Behavior," The science of health promotion, 1997.

[89] J. O. Prochaska and C. C. DiClemente, "Stages and processes of self-change of smoking: toward an integrative model of change," Journal of consulting and clinical psychology, vol. 51, p. 390, 1983.

[90] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," Proceedings of the IEEE, vol. 77, pp. 257-286, 1989.

[91] P. Robinson, "Task complexity, task difficulty, and task production: Exploring interactions in a componential framework," Applied Linguistics, vol. 22, p. 27, 2001.

[92] Jahmm, HMM implementation in Java, http://www.run.montefiore.ulg.ac.be/~franco

is/software/jahmm.

[93] Robinson, W.N., Syed A., Akhlaghi A., and Deng T. "Pattern Discovery of User Interface Sequencing by Rehabilitation Clients with Cognitive Impairments," in Hawaii International Conference on Software Systems, HI, USA, 2012.

[94] Schunk, "Learning Theories: An Educational Perspective" 1996, p.2.

[95] Morris, "Learning Theories for Teachers" 1971, p. 244.

[96] Dressel, "On Teaching and Learning in College" 1982, p. 21.

[97] American Association for Higher Education, 1998.

[98] S. Kullback, Information theory and statistics: Dover Pubns, 1997.

[99] Robinson, W.N. "Monitoring Software Requirements using Instrumented Code," IEEE Proceedings of The 35th Annual Hawaii International Conference on Systems Sciences, Hawaii, 2002.

[100] Robinson, W.N., Akhlaghi A., Syed A., and Deng T. "Discovery and Diagnosis of Behavioral Transitions in Rehabilitation Patient Event-Streams," ACM Transactions on Management Information Systems (TMIS), 2012.

[101] Robinson, W.N., Akhlaghi A., and Deng T. "Transition Discovery of Sequential Behaviors in Email Application Usage Using Hidden Markov Models," in Hawaii International Conference on Software Systems, HI, USA, 2013.

[102] Robinson, W.N., Deng T., and Akhlaghi A. "Transition Discovery in Open Source Development from a Distributive Cognition Perspective" in Hawaii International Conference on Software Systems, HI, USA, 2014.

[103] R. Schwarzer, "Modeling health behavior change: How to predict and modify the adoption

and maintenance of health behaviors," Applied Psychology, vol. 57, pp. 1-29, 2008.

[104] Spanoudakis, G., Kloukinas, C., Tsigritis, T., Androutsopoulos, K., Ballas, C., and Presenza, D. "Review of the state of the art in Security and Dependability Monitoring and Recovery," Report no. A4.D1.1, EU F6 Programme, SERENITY Project, London.

[105] A. Srivastava, A. Kundu, S. Sural, and A. K. Majumdar, "Credit card fraud detection using hidden Markov model," Dependable and Secure Computing, IEEE Transactions on, vol. 5, pp. 37-48, 2008.

[106] Van der Aalst, W., and Weijters, A. "Process mining: a research agenda," Computers in Industry (53:3) 2004, pp 231-244.

[107] W. M. P. van der Aalst, et al., "Workflow mining: a survey of issues and approaches," Data & Knowledge Engineering, vol. 47, pp. 237-267, 2003.

[108] van Lamsweerde, A. "From System Goals to Software Architecture," Formal Methods for Software Architectures) 2003, pp 25–43.

[109] van Lamsweerde, A. "Elaborating security requirements by construction of intentional anti-models," Software Engineering, 2004. ICSE 2004. Proceedings. 26th International Conference on) 2004, pp 148-157.

[110] van Lamsweerde, A., Darimont, R., and Massonet, P. "Goal-Directed Elaboration of Requirements for a Meeting Scheduler: Problems and Lessons Learnt," IEEE, Second International Symposium on Requirements Engineering, 1995, pp. pp. 194-203.

[111] Virone, G., Alwan, M., Dalal, S., Kell, S.W., Turner, B., Stankovic, J.A., and Felder, R. "Behavioral Patterns of Older Adults in Assisted Living," Information Technology in Biomedicine, IEEE Transactions on (12:3) 2008, pp 387-398.

[112] Wang, Y., Yu, Y., and Mylopoulos, J. "An Automated Approach for Monitoring and Diagnosing Requirements," The 22nd IEEE/ACM International Conference on Automated Software Engineering, Atlanta, GA, 2007, pp. 293-302.

[113] J. T. L. Wang, et al., "Combinatorial pattern discovery for scientific data: Some preliminary results," 1994, pp. 115-125.

[114] Watson, H., Wixom, B., Hoffer, J., Anderson-Lehman, R., and Reynolds, A. "Real-time business intelligence: Best practices at Continental Airlines," Information Systems Management (23:1) 2006, pp 7-18.

[115] Wilson, B.A., Emslie, H.C., Quirk, K., and Evans, J.J. "Reducing everyday memory and planning problems by means of a paging system: a randomised control crossover study," Journal of Neurology, Neurosurgery, and Psychiatry (70:4) 2001, pp 477-482.

[116] Wright, P., Rogers, N., Hall, C., Wilson, B., Evans, J., Emslie, H., and Bartram, C. "Comparison of pocket-computer memory aids for people with brain injury," Brain Injury (15:9) 2001, pp 787-800.

[117] Zeng, Y., Chiang, R., and Yen, D. "Enterprise integration with advanced information technologies: ERP and data warehousing," Information Management & Computer Security (11:3) 2003, pp 115-122.

[118] Zhang, J., Cheng, B., and Goldsby, H. "AMOEBA-RT: Run-Time Verification of Adaptive Software," Models@Run.Time'07 Workshop, ACM/IEEE, ACM/IEEE 10th International MoDELS 2007, Nashville, TN, 2007.

[119] Q. Zhao and S. S. Bhowmick, "Sequential pattern mining: A survey," ITechnical Report CAIS Nayang Technological University Singapore, pp. 1–26, 2003.