Noname manuscript No.
(will be inserted by the editor)

A Reference Based Analysis Framework for Understanding Anomaly Detection Techniques for Symbolic Sequences

Varun Chandola · Varun Mithal · Vipin Kumar

Received: date / Accepted: date

Abstract Anomaly detection for symbolic sequence data is a highly important area of research and is relevant in many application domains. While several techniques have been proposed within different domains, understanding of their relative strengths and weaknesses is limited. The key factor for this is that the nature of sequence data varies significantly across domains, and hence while a technique might perform well in its original domain, its performance is not guaranteed in a different domain. In this paper, we aim at establishing this understanding for a wide variety of anomaly detection techniques for symbolic sequences. We present a comparative evaluation of a large number of anomaly detection techniques on a variety of publicly available as well as artificially generated data sets. Many of these are existing techniques while some are slight variants and/or adaptations of traditional anomaly detection techniques to sequence data. The analysis presented in this paper allows relative comparison of the different anomaly detection techniques and highlights their strengths and weaknesses. We extend the Reference Based Analysis (RBA) framework, which was originally proposed to analyze multivariate categorical data, to analyze symbolic sequence data sets. We visualize the symbolic sequences using the characteristics provided by the RBA framework and use the visualization to understand various aspects of the sequence data. We then use the characterization done by RBA to understand the performance of the

This work was supported by NASA under award NNX08AC36A, NSF Grant CNS-0551551 and NSF Grant IIS-0713227. Access to computing facilities was provided by the Digital Technology Consortium.

V. Chandola
Geographic Information Science and Technology
Oak Ridge National Laboratory
E-mail: chandolav@ornl.gov

V. Mithal, V. Kumar
Department of Computer Science
University of Minnesota
E-mail: vmithal,kumar@cs.umn.edu

different techniques. Using the RBA framework, we propose two anomaly detection techniques for symbolic sequences, which show consistently superior performance over the existing techniques across the different data sets. We also propose a novel artificial data generator that can be used to generate validation data sets to evaluate anomaly detection techniques for sequences. The generator allows creation of validation data sets with different characteristics by varying the associated parameters to study the relationship between the anomaly detection techniques and the different characteristics of the data.

1 Introduction

Data occurs naturally as sequences in a wide variety of applications, such as system call logs in a computer, biological sequences, operational logs of an aircrafts' flight, etc. In several such domains, anomaly detection is required to detect events of interests as *anomalies*. There has been extensive research done on anomaly detection techniques [1–3], but most of these techniques assign an anomaly score to individual data instances with respect to the the normal data instances, without accounting for the sequence aspect of the data. For example, consider the set of user command sequences shown in Table 1. Clearly the sequence S_5 is anomalous, corresponding to a hacker breaking into a computer after multiple failed attempts, even though each command in the sequence by itself is normal.

S_1	$login, pwd, mail, ssh, \ldots, mail, web, logout$
S_2	$login, pwd, mail, web, \ldots, web, web, web, logout$
S_3	$login, pwd, mail, ssh, \ldots, mail, web, web, logout$
S_4	$login, pwd, web, mail, ssh, \ldots, web, mail, logout$
S_5	$\mathbf{login}, \mathbf{pwd}, \mathbf{login}, \mathbf{pwd}, \mathbf{login}, \mathbf{pwd}, \ldots, \mathbf{logout}$

Table 1 Sequences of User Commands

A large number of anomaly detection techniques for symbolic sequences have been proposed as shown in Table 2. The techniques can be classified into three broad categories based on the underlying approach. *Kernel based* techniques assign an anomaly score to a test sequence based on its similarity to the normal sequences. *Window based* techniques calculate the probability of occurrence of every fixed length window in the test sequence. *Markovian* techniques calculate the probability of occurrence of each symbol in the test sequence conditioned on the preceding few symbols in the test sequence. As Table 2 shows, these techniques have been developed in the context of different domains. For example, Sun et al [4] proposed a probabilistic suffix trees (PST) based technique to detect anomalous sequences in a data base of protein sequences. Forrest et al proposed several techniques to detect anomalous sequences in a data base of operating system call sequences [5–7]. While the

techniques were proposed and evaluated in specific domains, no systematic evaluation is available regarding their relative performance. In particular, it is unclear if the techniques are the best ones for the domain they were proposed for or if another technique (originally proposed for an entire different domain) might perform better.

In our previous work [8], we provided an experimental evaluation of the techniques listed in Table 2 on a variety of publicly available data sets, collected from different application domains, as well as artifical data sets, created using an artificial data generator. The results on different data sets reveal that no one technique is clearly superior to others. Most techniques show consistency in performance on public data sets belonging to one domain but show different performance for data sets from a different domain. This indicates a relationship between the techniques and the nature of the data. These observations motivate a deeper study of the relationship between a technique and a data set.

In this paper we build upon our previous work [8] to further understand the relationship between the anomaly detection techniques and the nature of data. To gain this understanding we use an analysis methodology, called *Reference Based Analysis* (RBA) [9]. RBA was originally proposed to characterize and visualize multivariate categorical data sets. In this paper we show how the same framework can be adapted to characterize symbolic sequence data. We visualize the symbolic sequences using these characteristics and demonstrate its utility in understanding various aspects of the sequence data such as how different are the normal sequences from the anomalous sequences and how similar are the normal sequences to each other. We then show how different anomaly detection techniques listed in Table 2 rely on one or more of such characteristics to detect anomalies. Using these characteristics, we propose two novel anomaly detection techniques for symbolic sequences, called WIN_{1D} and WIN_{2D}, which show consistently superior performance over the existing techniques across the different data sets.

Application Domains	Kernel Based Techniques	Window Based Techniques	Markovian Techniques		
			Fixed	Variable	Sparse
Intrusion Detection		[5],[6], [7],[10]	[7],[11], [12],[13], [14],[15]		[7], [16]
Proteomics				[4]	
Flight Safety	[17]		[18]		

Table 2 Anomaly Detection Techniques for Symbolic Sequences.

1.1 Our Contributions

The specific contributions of this paper are as follows:

- We provide an experimental evaluation of anomaly detection techniques listed in Table 2 on a variety of data sets to explain the performance of a variety of anomaly detection techniques on different types of sequence data sets. The analysis presented in this paper allows relative comparison of the different anomaly detection techniques and highlights their strengths and weaknesses. A preliminary version of the results were published in [8] in which the techniques were evaluated using the *precision on anomaly class* metric. In this paper we also present results using the *area under ROC curve* metric.
- We extend the RBA framework [9], which was originally proposed to analyze multivariate categorical data, to analyze symbolic sequence data sets. We visualize the symbolic sequences using the characteristics provided by the RBA framework and use the visualization to understand various aspects of the sequence data. We then use the characterization done by RBA to understand the performance of the different techniques listed in Table 2. A preliminary version of this work has been presented in the context of data sets collected from the domain of system call intrusion detection [19].
- Using the RBA framework, we propose two anomaly detection techniques for symbolic sequences, called WIN_{1D} and WIN_{2D}, which show consistently superior performance over the existing techniques across the different data sets.
- We propose a novel artificial data generator that can be used to generate validation data sets to evaluate anomaly detection techniques for sequences. The generator allows creation of validation data sets with different characteristics by varying the associated parameters to study the relationship between the anomaly detection techniques and the different characteristics of the data.

1.2 Organization

The rest of this paper is organized in two parts. In the first part (Sections 2–6), we provide a description of the different anomaly detection techniques listed in Table 2 and an experimental comparison of their performance on a variety of publicly available as well as synthetic data sets. In the second part (Sections 7–3) we show how the RBA framework can be used to analyze symbolic sequences. Specifically, we show how the RBA framework can be used understand the relationship between different anomaly detection techniques and the nature of sequence data in Sections. In Section 8, we show how the RBA framework can be used to analyze symbolic sequences. Specifically, we show how the RBA framework can be used understand the relationship between different anomaly detection techniques and the nature of sequence data in Sections 9 and 10. In Section 11 we present two novel RBA based anomaly detection techniques for symbolic sequences.

2 Problem Statement

The objective of the techniques evaluated in this paper can be stated as follows:

Definition 1 *Given a set of n training sequences, $\mathbf{T}$, and a set of m test sequences $\mathbf{S}$, find the anomaly score $A(S_i)$ for each test sequence $S_i \in \mathbf{S}$, with respect to $\mathbf{T}$.*

All sequences consist of events that correspond to a finite alphabet, Σ. The length of sequences in $\mathbf{T}$ and sequences in $\mathbf{S}$ might or might not be equal in length. The training database $\mathbf{T}$ is assumed to contain only normal sequences, and hence the techniques operate in a semi-supervised setting [20].

*Notation*In this paper we will denote training sequences as T or T_j and test sequences as S or S_i. The size of the training data set is denoted by m and size of the test data set is denoted by n.

3 Anomaly Detection Techniques for Sequences

We evaluated a variety of techniques that can be grouped into following three categories:

3.1 Kernel Based Techniques

Kernel based techniques make use of kernel K by using the pairwise similarity between sequences. In the problem formulation stated in Definition 1 the sequences can be of different lengths, hence simple measures such as *Hamming Distance* cannot be used. One possible measure is the normalized length of *longest common subsequence* between a pair of sequences. This similarity between two sequences S_1 and S_2, is computed as:

$$nLCS(S_1, S_2) = \frac{|LCS(S_1, S_2)|}{\sqrt{|S_1||S_2|}} \tag{1}$$

Since the value computed above is between 0 and 1, the distance between S_1 and S_2 can be computed as [21]:

$$d(S_1, S_2) = 1 - nLCS(S_1, S_2) \tag{2}$$

Other similarity measures other than $nLCS$ can be used as well, e.g., the *spectrum kernel* [22] or time series bitmaps [23]. We use $nLCS$ in our experimental study, since it was used in [17] to detect anomalies in discrete sequences and appears promising.

*Computing Kernel*For a given test data set, a kernel matrix $K \in \Re^{m \times n}$ is computed such that:

$$K[i][j] = nLCS(S_i, S_j),\ S_i \in \mathbf{S},\ S_j \in \mathbf{T} \tag{3}$$

3.1.1 Nearest Neighbors Based (kNN)

In the nearest neighbor scheme (kNN), for each test sequence $S_i \in \mathbf{S}$, the distance to its k^{th} nearest neighbor in the training set $\mathbf{T}$ is computed using the kernel matrix K. This distance becomes the anomaly score $A(S_i)$ [21,24].

3.1.2 Clustering Based ($CLUSTER$)

This technique clusters the sequences in $\mathbf{T}$ into a fixed number of clusters, c, by using the kernel matrix K. The test phase involves measuring the distance of every test sequence, $S_i \in \mathbf{S}$, with the medoid of each $CLUSTER$. The distance to the medoid of the closest $CLUSTER$ becomes the anomaly score $A(S_i)$.

3.2 Window Based Techniques

Window based techniques try to localize the cause of anomaly in a test sequence, within one or more windows, where a window is a fixed length subsequence of the test sequence. One such technique called *Threshold Sequence Time-Delay Embedding* ($tSTIDE$) [7] uses a sliding window of fixed size k to extract k-length windows from the training sequences in $\mathbf{T}$. The count of each window occurring in $\mathbf{T}$ is maintained. During testing, k-length windows are extracted from a test sequence S_i. Each such window ω_i is assigned a likelihood score $P(\omega_i) = \frac{f(\omega_i)}{f(*)}$, where $f(\omega_i)$ is the frequency of occurrence of window ω_i in $\mathbf{T}$, and $f(*)$ is the total number of k length windows extracted from $\mathbf{T}$.

For the test sequence S_i, $|S_i|-k+1$ windows are extracted, and a likelihood score vector of length $|S_i|-k+1$ is obtained. This score vector is then combined to obtain the anomaly score for the sequence, $A(S_i)$, in the following way:

$$L(S_i) = \frac{1}{|S_i| - k + 1} \sum_{i=1}^{|S_i|-k+1} \log P(\omega_i) \tag{4}$$

$$A(S_i) = -1 * L(S_i) \tag{5}$$

If likelihood score for any window is 0, it is replaced with 10^{-6} since $\log 0$ is undefined. Other alternatives to combine the score vector to obtain $A(S_i)$ are discussed in Section 3.4.

3.3 Markovian Techniques

Such techniques estimate the conditional probability for each symbol in a test sequence S_i conditioned on the symbols preceding it. Most of the techniques utilize the *short memory* property of sequences [25]. This property is a higher-order Markov condition which states that for a given sequence

$S = \langle s_1, s_2, \ldots s_{|S|} \rangle$, the conditional probability of occurrence of a symbol s_i is given as:

$$P(s_i|s_1 s_2 \ldots s_{i-1}) \approx P(s_i|s_{i-k+1} \ldots s_{i-1}), i > k \tag{6}$$

In the following, we investigate four Markovian techniques. Each one of them computes a vector of scores, each element of which corresponds to the conditional probability of observing a symbol, as defined in Equation 6. This score vector is then combined to obtain $A(S_i)$ using equations similar to Equations 4 and 5, by replacing $P(\omega_i)$ with $P(s_i|s_{i-k+1} \ldots s_{i-1})$.

3.3.1 Fixed Length Markovian Technique

A fixed length Markovian technique determines the probability $P(s_{qi})$ of a symbol s_{qi}, conditioned on a fixed number of preceding symbols[1]. One such technique uses an extended Finite State Automaton (FSA) to estimate the conditional probabilities. We will refer to this technique as FSA in subsequent discussions.

FSA extracts $(n+1)$ sized subsequences from the training data $\mathbf{T}$ using a sliding window. Each node in the automaton constructed by FSA corresponds to a unique subsequence of n symbols that form the first n symbols of such $n+1$ length subsequences. An edge exists between a pair of nodes, N_i and N_j in the FSA, if N_i corresponds to states $s_{i1}s_{i2} \ldots s_{in}$ and N_j corresponds to states $s_{i2}s_{i3} \ldots s_{in}s_{jn}$. At every state of the FSA two quantities are maintained. One is the number of times the n length subsequence corresponding to the state is observed in $\mathbf{T}$. The second quantity is a vector of frequencies corresponding to number of times different edges emanating from this state are observed. Using these two quantities, the conditional probability for a symbol, given preceding n symbols, can be determined.

During testing, the automaton is used to determine a likelihood score for every $n+1$ subsequence extracted from test sequence S_i which is equal to the conditional probability associated with the transition from the state corresponding to first n symbols to the state corresponding to the last n symbols. If there is no state in the automaton corresponding to the first n symbols, the subsequence is ignored.

$FSAz$We propose a variant of FSA technique, in which if there is no state corresponding to the first n symbols of a $(n+1)$ subsequence, we assign a low score (e.g. 0) to that subsequence, instead of ignoring it. The intuition behind assigning a low score to non-existent states is that anomalous test sequences are more likely to contain such states, than normal test sequences. While FSA ignores this information, we utilize it in $FSAz$.

[1] A more general formulation that determines probability of l symbols conditioned on a fixed number of preceding n symbols is discussed in [15].

3.3.2 Probabilistic Suffix Trees (PST)

A PST is a compact tree representation of a variable Markov chain, which uses classical *suffix trees* as its index structure [25]. We evaluate a PST based anomaly detection technique [4], that learns a PST from the training sequences and then assigns a conditional likelihood score to each symbol of the test sequence.

In a PST, each edge is labeled using a symbol, and each node represents the subsequence obtained by traversing the path from root to the node, as well as the number of times the subsequence is observed in the training sequences. Each node also stores the conditional probability of observing each symbol in the alphabet, given the subsequence represented by the node. The PST is grown (training phase) by scanning the training sequences. The maximum depth of the tree is fixed at k, which is a user defined parameter. Several pruning criterion are applied to the PST to ensure that the PST contains only those paths that occur significantly enough number of times in the training sequences. The pruning can be done by applying thresholds to the frequency of a node label, or to the conditional probability of a symbol emanating from a given node. If no pruning is applied, the PST is equivalent to the FSA-z.

During the testing phase for the PST based technique the test sequence is scanned and the PST is traversed simultaneously. For a symbol s_{qi} in the test sequence S_i, its conditional probability is estimated by finding the longest suffix of the k length subsequence that precedes s_{qi} (in S_i) and occurs as a path in the PST. Thus, different symbols are conditioned on a different sized history.

3.3.3 Sparse Markovian Technique

Sparse Markovian techniques are more flexible than variable Markovian techniques, in the sense that they estimate the conditional probability of s_{qi} based on a subset of symbols within the preceding k symbols, which are not necessarily contagious to s_{qi}. In other words the symbols are conditioned on a sparse history.

Lee et al. [13] use $RIPPER$ classifier to build one such sparse model. In this approach, a sliding window is applied to the training data $\mathbf{T}$ to obtain k length windows. The first $k-1$ positions of these windows are treated as $k-1$ categorical attributes, and the k^{th} position is treated as a target class. $RIPPER$ [26] is used to learn rules that can predict the k^{th} symbol given the first $k-1$ symbols. To ensure that there is no symbol that occurs very rarely as the target class, the training sequences are duplicated 5 times.

For testing, k length subsequences are extracted from each test sequence S_i using a sliding window. For any subsequence, the first $k-1$ events are classified using the classifier learnt in the training phase and the prediction is compared to the k^{th} symbol. $RIPPER$ also assigns a confidence score associated with the classification, denoted as $conf(s_{qi}) = \frac{100T}{M}$, where M is the number of times the particular rule was fired in the training data, and T is the number

of times the rule gave correct prediction. Lee et al. [13] assign the likelihood score of symbol s_{qi} as follows:

- For a correct classification, $P(s_{qi}) = 1$.
- For a misclassification, $P(s_{qi}) = \frac{1}{conf(s_{qi})} = \frac{M}{100T}$.

3.3.4 Hidden Markov Models Based Technique (HMM)

Techniques that apply HMMs for modeling sequences, transform an input sequence from the symbol space to the hidden state space. The key assumption for the HMM based anomaly detection technique [7] is that the normal sequences can be effectively represented in the hidden state space, while anomalous sequences cannot be.

The training phase involves learning an HMM with σ hidden states, from the normal sequences in $\mathbf{T}$ using the *Baum Welch* algorithm.In the testing phase, the optimal hidden state sequence for the given input test sequence S_i is determined, using the *Viterbi* algorithm.For every pair of consecutive states, $\langle s^H_{qi}, s^H_{qi+1} \rangle$, in the optimal hidden state sequence, the state transition matrix provides a likelihood score for transitioning from s^H_{qi} to s^H_{qi+1}. Thus a likelihood score vector of length $|S_i| - 1$ is obtained.

3.4 Combining Scores

For each of the window based and Markovian techniques, a likelihood score vector is generated for a test sequence, S_i. A combination function is then applied to obtain a single anomaly score $A(S_i)$. In Section 3.2, we presented one such combination technique, average log score, which was originally used in the technique [4]. $L(S_i)$ can be computed in other ways, such as average score [14], minimum score, maximum score, and using a threshold [15, 7]. For the threshold method, a user defined threshold is employed to determine which scores in the likelihood score vector are anomalous. The number of such anomalous scores is the anomaly score $A(S_i)$ of the test sequence. Setting the threshold often requires experimenting with different possible values, and then choosing the best performing value.

4 Data Sets Used

In this section we describe various public as well as the artificially generated data sets that we used to evaluate the different anomaly detection techniques. We used public data sets that have been used earlier to evaluate sequence anomaly detection techniques. To further illustrate certain aspects of different techniques, we constructed different artificial data sets. The artificial data sets were constructed such that we can control the nature of normal as well as anomalous sequences and hence learn the relationship between the various techniques and the nature of the data.

For every data set, we first constructed a set of normal sequences, and a set of anomalous sequences. A sample of the normal sequences was used as training data for different techniques. A disjoint sample of normal sequences and a sample of anomalous sequences were added together to form the test data. The relative proportion of normal and anomalous sequences in the test data determined the "difficulty level" for that data set. We experimented with different ratios such as 1:1, 10:1 and 20:1 of normal and anomalous sequences. Results on data sets with other ratios are consistent in relative terms, although most techniques perform much better for the simplest data set that uses a ratio 1:1. Since in real sequences anomalies are rare, we report results when normal and anomalous sequences were in 20:1 ratio in test data. In reality, the ratio of normal to anomalous can be even larger than 20:1. But we were unable to try such skewed distributions due to limited number of normal samples available in some of the data sets.

Source	Data Set	$\|\Sigma\|$	$\hat{l}$	$\|\mathbf{S^N}\|$	$\|\mathbf{S^A}\|$	$\|\mathbf{T}\|$	$\|\mathbf{S}\|$
PFAM	HCV	44	87	2423	50	1423	1050
	NAD	42	160	2685	50	1685	1050
	TET	42	52	1952	50	952	1050
	RUB	42	182	1059	50	559	525
	RVP	46	95	1935	50	935	1050
UNM	snd-cert	56	803	1811	172	811	1050
	snd-unm	53	839	2030	130	1030	1050
DARPA	bsm-week1	67	149	1000	800	10	210
	bsm-week2	73	141	2000	1000	113	1050
	bsm-week3	78	143	2000	1000	67	1050

Table 3 Public data sets used for experimental evaluation. $\hat{l}$ – Average Length of Sequences, $\mathbf{S^N}$ – Normal Data, $\mathbf{S^A}$ – Anomalous Data, $\mathbf{T}$ – Training Data, $\mathbf{S}$ – Test Data.

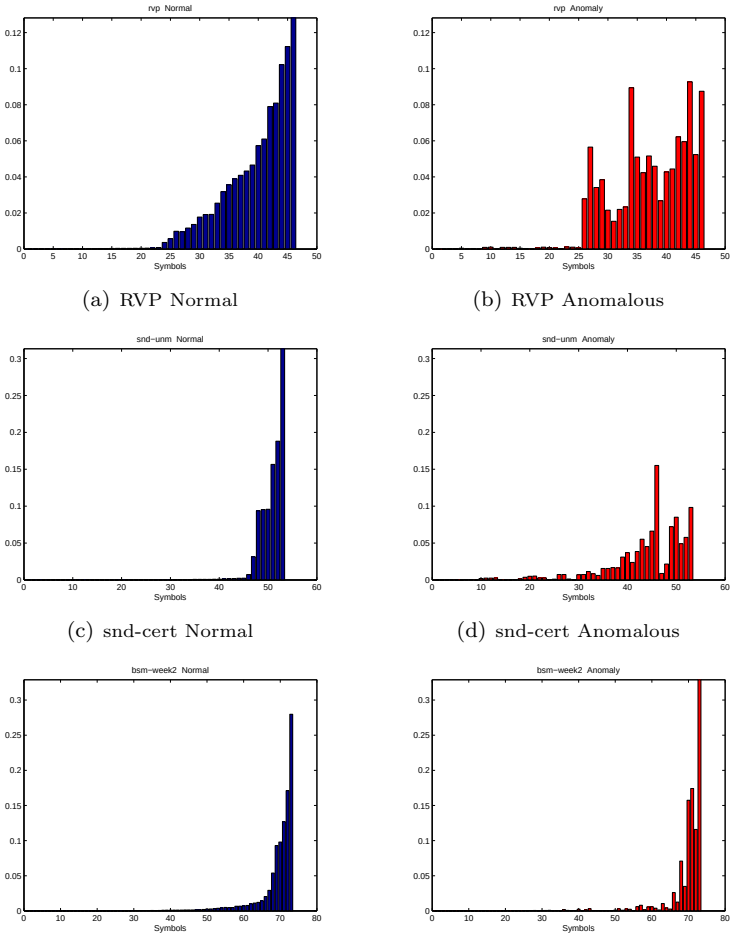
4.1 Public Data Sets

Table 3 summarizes the various statistics of the data sets used in our experiments. All data sets are available from our web site[2]. The distribution of the symbols for normal and anomalous sequences is illustrated in Figures 1(a),1(b) (RVP), 1(c),1(d) (snd-unm), and 1(e),1(f), (bsm-week2). The distribution of symbols in snd-unm data is different for normal and anomalous data, while the difference is not significant in RVP and bsm-week2 data. We will explain how the normal and anomalous sequences were obtained for each type of data set in the next subsections.

4.1.1 Protein Data Sets

The first set of public data sets were obtained from PFAM database (Release 17.0) [27] containing sequences belonging to 7868 protein families. Sequences

[2] http://www.cs.umn.edu/ chandola/ICDM2008



(a) RVP Normal (b) RVP Anomalous

(c) snd-cert Normal (d) snd-cert Anomalous

(e) bsm-week2 Normal (f) bsm-week2 Anomalous

Fig. 1 Distribution of Symbols in Training Data Sets of Different Types.

belonging to one family are structurally different from sequences belonging to another family. We choose five families, viz., HCV, NAD, TET, RVP, RUB. For each family we construct a normal data set by choosing a sample from the set of sequences belonging to that family. We then sample 50 sequences from other four families to construct an anomaly data set. Similar data was used by [4] to evaluate the PST technique. The difference was that the authors constructed a test data for each pair of protein families such that samples from one family were used as normal and samples from the other were used as test. The PST results on PFAM data sets reported in this paper appear to be worse than those reported in [4].

4.1.2 Intrusion Detection Data Sets

The second set of public data sets were collected from two repositories of benchmark data generated for evaluation of intrusion detection algorithms. One repository was generated at University of New Mexico[3]. The normal sequences consisted of sequence of system calls generated in an operating system during the normal operation of a computer program, such as sendmail, ftp, lpr etc. The anomalous sequences consisted of sequence of system calls generated when the program is run in an abnormal mode, corresponding to the operation of a hacked computer. We report results on two data sets, viz, *snd-unm* and *snd-cert*. Other data sets were not used due to insufficient anomalous sequences to attain a 20:1 imbalance. For each of the two data sets, the number of sequences in the normal as well as anomaly data was small (less than 200), making it difficult to construct significant test and training data sets. The increase the size of the data sets, we extracted subsequences of length 100 by sliding a window of length 100 and a sliding step of 50. The subsequences extracted from the original normal sequences were treated as normal sequences and the subsequences extracted from the original anomalous sequences were treated as anomalous sequences if they did not occur in the normal sequences.

The other intrusion detection data repository was the *Basic Security Module* (BSM) audit data, collected from a victim Solaris machine, in the DARPA Lincoln Labs 1998 network simulation data sets [28]. The repository contains labeled training and testing DARPA data for multiple weeks collected on a single machine. For each week we constructed the normal data set using the sequences labeled as normal from all days of the week. The anomaly data set was constructed in a similar fashion. The data is similar to the system call data described above with similar (though larger) alphabet.

4.2 Altered RVP Data Set

To better understand the performance of the anomaly detection techniques to the nature of anomalies in the test data, we created a data set from the original RVP data from the PFAM repository. A test data set was constructed by sampling 800 most normal sequences not present in training data. Anomalies were injected in 50 of the test sequences by randomly replacing k symbols in each sequence with the least frequent symbol in the data set. The parameter k controls the deviation of the anomalous sequences from the normal sequences. The objective of this experiment was to evaluate how the performance of a technique varies with k.

[3] http://www.cs.unm.edu/∼immsec/systemcalls.htm

4.3 Artificial Data Sets

As mentioned in the introduction, two types of anomalous sequences can exist, one which are arguably generated from a different generative mechanism than the normal sequences, and the other which result from a normal sequence deviating for a short span from its expected normal behavior. To study the relationship between these two types of anomalous sequences and the performance of different techniques, we designed an artificial data generator which allows us to generate validation data sets with different types of anomalies.

We used a generic HMM, as shown in Figure 2 to model normal as well as anomalous data. The HMM shown in Figure 2 has two sets of states,

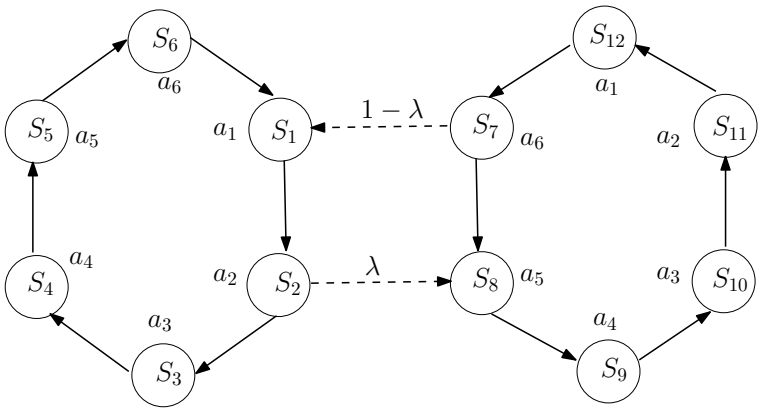


Fig. 2 HMM used to generate artificial data.

$\{S_1, S_2, \ldots S_6\}$ and $\{S_7, S_8, \ldots S_{12}\}$.

Within each set, the transitions corresponding to the solid arrows shown in Figure 2 were assigned a transition probability of $(1-5\beta)$, while other transitions were assigned transition probability β. No transition is possible between states belonging to different sets. The only exception are S_2S_8 for which the transition probability is λ, and S_7S_1 for which the transition probability is $1-\lambda$. The transition probabilities S_2S_3 and S_7S_8 are adjusted accordingly so that the sum of transition probabilities for each state is 1.

The observation alphabet is of size 6. Each state emits one alphabet with a high probability $(1-5\alpha)$, and all other alphabets with a low probability (α). Figure 2 depicts the most likely alphabet for each state.

The initial probability vector π of the HMM is constructed such that either $\pi_1 = \pi_2 = \ldots = \pi_6 = 1$ and $\pi_7 = \pi_8 = \ldots = \pi_{12} = 0$; or vice-versa.

Normal sequences are generated by setting λ to a low value and π to be such that the first 6 states have initial probability set to $\frac{1}{6}$ and rest 0. If $\lambda = \beta = \alpha = 0$, the normal sequences will consist of the subsequence $a_1a_2a_3a_4a_5a_6$ getting repeated multiple times. By increasing λ or β or α, anomalies can be induced in the normal sequences.

This generic HMM can be tuned to generate two types of anomalous sequences. For the first type of anomalous sequences, λ is set to a high value and π to be such that the last 6 states have initial probability set to $\frac{1}{6}$ and rest 0. The resulting HMM is directly opposite to the HMM constructed for generating normal sequences. Hence the anomalous sequences generated by this HMM are completely different from the normal sequences.

To generate second type of anomalous sequences, the HMM used to generate the normal sequence is used, with the only difference that λ is increased to a higher value than 0. Thus the anomalous sequences generated by this HMM will be similar to the normal sequences except that there will be short spans when the symbols are generated by the second set of states.

By varying λ, β, and α, we generated several evaluation data sets (with two different type of anomalous sequences). We will present the results of our experiments on these artificial data sets in next section.

5 Evaluation Methodology

The techniques investigated in this paper assign an anomaly score to each test sequence $S_i \in \mathbf{S}$, such that the sequence with highest anomaly score is considered as most anomalous, and so on. To compare the performance of different techniques in such a scenario, we first convert the ranked output for each sequence in the test data set into a binary label (normal or anomalous) in the following manner:

1. Rank the test sequences in decreasing order based on the anomaly scores.
2. Label the sequences in the top p portion of the sorted test sequences as anomalous, and rest sequences as normal, where $1 < p \leq |S|$.

We evaluate the techniques using two different metrics. The first metric uses a fixed value of p. Let there be t true anomalous sequences in top p ranked sequences. We measure the accuracy of the techniques as:

$$Accuracy = \frac{t}{p}$$

We report the accuracy when $p = q$, where q is number of true anomalous sequences in the test data set. The accuracies for other values of p also showed consistent results.

One drawback of the above metric is that it is highly dependent on the choice of p. A technique might show 100% accuracy for a particular value of p but show 50% accuracy for $2p$. To overcome this drawback we use a second evaluation metric.

The second metric used to evaluate the anomaly detection techniques is to obtain the area under the ROC curve (AUC) obtained by varying p from 1 to $|S|$. The advantage of AUC is that it is not dependent on the choice of p.

6 Experimental Results

The experiments were conducted on a variety of data sets discussed in Section 4. The various parameter settings associated with each technique were explored. The results presented here are for the parameter setting which gave best results across all data sets, for each technique.

6.1 Sensitivity to Parameters

The performance of $CLUSTER$ improved as c was increased from 2 onwards, but stabilized for values greater than 32. The best overall performance was observed for $c = 32$. For kNN, the performance was comparable for a wide range of k ($2 \leq k \leq 32$) but deteriorated for higher values of k. The best overall performance was observed for $k = 4$. For $tSTIDE$ as well as the Markovian techniques (FSA, $FSAz$, PST, $RIPPER$), the performance was sensitive to the choice of window length or the length of the history. For low values of this length (≤ 5) or for values higher than 10, the performance was generally poor. The best performing setting was window size of 6 for $tSTIDE$ and history length of 5 for the Markovian techniques. For PST, an additional parameter is P_{min} which controls the threshold under which the counts for a given subsequence are considered insignificant. We observed that performance of PST was highly sensitive to this parameter. If P_{min} was set to very low (≈ 0), PST performed similar to $FSAz$, while if P_{min} was set to be higher than 0.1, the performance was poor. The best performance of PST was observed for $P_{min} = 0.01$. For HMM, the number of hidden states σ is a critical parameter. We experimented with values ranging from 2 to $|\Sigma|$. Our experiments reveal that the performance of HMM does not vary significantly for different values of σ. The best overall performance of HMM was observed for $\sigma = 4$ for public data sets and $\sigma = 12$ for the artificial data sets.

We experimented with various combination functions for different techniques, and found that the *average log score* function has the best performance across all data sets. Hence, results are reported for the *average log score* function. Results with other combination techniques are available in our technical report [29].

6.2 Accuracy vs. AUC

We evaluated the different techniques using the two evaluation metrics described in Section 5, *Accuracy* and *AUC*. Both metrics show similar relative performance for the different techniques. We will compare the performance using the *accuracy* metric.

6.3 Results on Public Data Sets

Tables 4 and 5 summarize the accuracy and AUC results on the 10 public data sets. $CLUSTER$ and kNN show good performance for PFAM and UNM data sets but perform moderately on DARPA data sets. FSA and $FSAz$ show consistently good performance for all public data sets. $tSTIDE$ performs well for PFAM data sets but its performance degrades for both UNM and DARPA data sets. PST performs average to poor for all data sets including the PFAM data sets for which it was originally used. The HMM technique performs poorly for all public data sets. The reasons for the poor performance is that HMM technique makes an assumption that the normal sequences can be represented with σ hidden states, which might not be true for the public data sets.

	PFAM					UNM		DARPA			
	hcv	nad	tet	rvp	rub	snd-unm	snd-cert	bsm-week1	bsm-week2	bsm-week3	**Avg**
cls	0.54	0.46	0.84	0.86	0.76	0.76	**0.94**	0.20	0.36	0.52	0.62
knn	0.88	0.64	**0.86**	**0.90**	0.72	**0.84**	**0.94**	0.20	0.52	0.48	0.70
tstd	0.90	**0.74**	0.50	**0.90**	**0.88**	0.58	0.64	0.20	0.36	0.60	0.63
fsa	0.88	0.66	0.48	**0.90**	0.80	0.82	0.88	0.40	0.52	0.64	0.70
fsaz	**0.92**	0.72	0.50	**0.90**	**0.88**	0.80	0.88	**0.50**	**0.56**	**0.66**	**0.73**
pst	0.74	0.10	0.66	0.50	0.28	0.28	0.10	0.00	0.10	0.34	0.31
rip	0.52	0.20	0.36	0.66	0.72	0.72	0.70	0.20	0.18	0.50	0.48
hmm	0.10	0.06	0.20	0.10	0.00	0.00	0.00	0.00	0.02	0.20	0.07
Avg	0.69	0.45	0.55	0.72	0.63	0.60	0.64	0.21	0.33	0.49	

Table 4 Accuracy results for public data sets.

	PFAM					UNM		DARPA			
	hcv	nad	tet	rvp	rub	snd-unm	snd-cert	bsm-week1	bsm-week2	bsm-week3	**Avg**
cls	0.98	0.96	1.00	1.00	0.99	0.99	1.00	0.74	0.90	0.91	0.94
knn	1.00	0.98	1.00	1.00	0.99	1.00	1.00	0.75	0.92	0.91	0.95
tstd	0.99	0.97	0.98	1.00	1.00	0.97	0.92	0.62	0.73	0.80	0.90
fsa	0.98	0.97	0.92	0.99	0.99	0.99	0.96	0.88	0.90	0.97	0.96
fsaz	1.00	0.98	0.98	1.00	1.00	0.97	0.96	0.88	0.91	0.97	0.96
pst	0.99	0.54	0.98	0.97	0.91	0.93	0.88	0.35	0.42	0.54	0.75
rip	0.70	0.45	0.37	0.97	0.96	0.98	0.94	0.79	0.70	0.84	0.77
hmm	0.58	0.50	0.71	0.55	0.24	0.04	0.03	0.43	0.50	0.77	0.43
Avg	0.90	0.79	0.87	0.93	0.88	0.86	0.84	0.68	0.75	0.84	

Table 5 AUC results for public data sets.

Overall, one can observe that the performance of techniques in general is better for PFAM data sets and on UNM data sets, while the DARPA data sets are more challenging.

6.4 Results on Altered RVP Data Set

Figure 3 shows the performance of the different techniques on the altered RVP data set, for different values of k from 1 to 10. We observe that $FSAz$ performs remarkably well for these values of k. $CLUSTER$, $tSTIDE$, FSA, PST, and $RIPPER$ exhibit moderate performance, though for values of k closer to 10, $RIPPER$ performs better than the other 4 techniques. For $k > 10$, all techniques show better than 90% accuracy because the anomalous sequences become very distinct from the normal sequences, and hence all techniques perform comparably well.

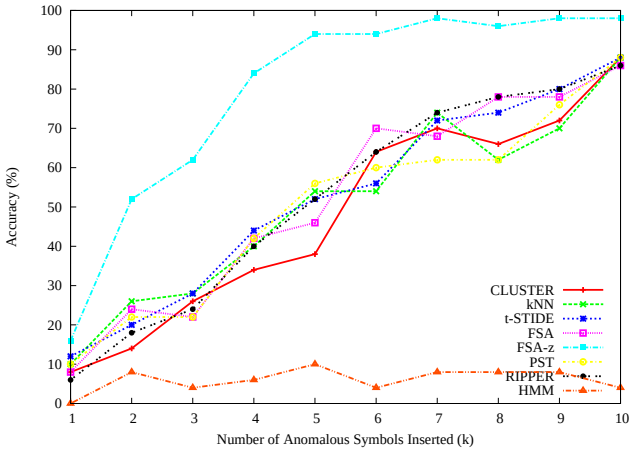


Fig. 3 Results for altered RVP data sets

6.5 Results on Artificial Data Sets

Tables 6 and 7 summarize the accuracy and AUC results on 6 ($d1$–$d6$) artificial data sets.The normal sequences in data set $d1$ were generated with $\lambda = 0.01, \beta = 0.01, \alpha = 0.01$. The anomalous sequences were generated using the first setting as discussed in Section 4.3, such that the sequences were primarily generated from the second set of states. For data sets $d2$–$d6$, the HMM used to generate normal sequences was tuned with $\beta = 0.01, \alpha = 0.01$. The value of λ was increased from 0.002 to 0.01 in increments of 0.002. The anomalous sequences for data sets $d2$–$d6$ were generated using the second setting in which λ is set to 0.1.

From Table 6, we observe that PST is the most stable technique across the artificial data sets, while the deterioration is most pronounced for FSA and $FSAz$. Both kNN and $CLUSTER$ also get negatively impacted as the λ increases but the trend is gradual than for $FSAz$. The performance of HMM

	d1	d2	d3	d4	d5	d6	**Avg**
cls	**1.00**	0.80	0.74	0.74	0.58	0.64	0.75
knn	**1.00**	0.88	0.76	**0.76**	0.60	**0.68**	0.78
tstd	**1.00**	0.82	0.64	0.64	0.48	0.50	0.68
fsa	**1.00**	0.88	0.50	0.52	0.24	0.28	0.57
fsaz	**1.00**	**0.92**	0.60	0.52	0.32	0.38	0.62
pst	**1.00**	0.84	**0.82**	**0.76**	**0.68**	**0.68**	**0.80**
rip	**1.00**	0.78	0.64	0.66	0.52	0.44	0.67
hmm	**1.00**	0.50	0.34	0.42	0.16	0.66	0.51
Avg	1.00	0.80	0.63	0.63	0.45	0.53	

Table 6 Accuracy results for artificial data sets.

	d1	d2	d3	d4	d5	d6	**Avg**
cls	1.00	0.95	0.97	0.98	0.94	0.95	0.97
knn	1.00	0.96	0.98	0.98	0.96	0.95	0.97
tstd	1.00	0.96	0.98	0.98	0.96	0.95	0.97
fsa	1.00	0.96	0.98	0.98	0.96	0.95	0.97
fsaz	1.00	0.96	0.98	0.98	0.96	0.95	0.97
pst	1.00	0.96	0.98	0.98	0.96	0.95	0.97
rip	1.00	0.96	0.98	0.98	0.96	0.95	0.97
hmm	1.00	0.96	0.98	0.98	0.96	0.95	0.97
Avg	1.00	0.96	0.98	0.98	0.96	0.95	

Table 7 AUC results for artificial data sets.

on the artificial data sets is better than for public data sets since the training data was actually generated by a 12 state HMM and the HMM technique was trained with $\sigma = 12$; thus the HMM model effectively captures the normal sequences.

6.6 Relative Performance of Different Techniques

Kernel based techniques are found to perform well for data sets in which the anomalous sequences are significantly different from the normal sequences; but perform poorly when the different between the two is small. This is due to the nature of the normalized LCS similarity measure used in the kernel based techniques. Our experiments show that kNN technique is somewhat better suited than $CLUSTER$ for anomaly detection, which is expected, since kNN is optimized to detect anomalies while the clustering algorithm in $CLUSTER$ is optimized to obtain clusters in the data.

$FSAz$ is consistently superior among all techniques, especially for data sets in which the anomalous sequences are minor deviations from normal sequences. The performance of $FSAz$ is poor when the normal sequences contain rare patterns. $FSAz$ is consistently superior to FSA. Performance of $tSTIDE$ is comparable to $FSAz$ when the anomalous sequences are significantly different from the normal sequences, but is inferior to $FSAz$ when the difference is small. $tSTIDE$ is less affected by the presence of rare patterns in the normal sequences than $FSAz$. for all PFAM data sets but is relatively poor on

DARPA and UNM data sets. $tSTIDE$ performs significantly better on artificial data sets. PST performs relatively worse than other techniques, except for cases where the normal sequences themselves contain many rare patterns. $RIPPER$ is also an average performer on most of the data sets, and is relatively better than PST, indicating that using a sparse history model is better than a variable history model.

For the public data sets, we found the HMM technique to perform poorly. The reasons for the poor performance of HMM are twofold. The first reason is that HMM technique makes an assumption that the normal sequences can be represented with σ hidden states. Often, this assumption does not hold true, and hence the HMM model learnt from the training sequences cannot emit the normal sequences with high confidence. Thus all test sequences (normal and anomalous) are assigned a low probability score. The second reason for the poor performance is the manner in which a score is assigned to a test sequence. The test sequence is first converted to a hidden state sequence, and then a $(1 + 1)$ FSA is applied to the transformed sequence. We have observed from our experiment using FSA that a $(1+1)$ FSA does not perform well for anomaly detection. The performance of HMM on artificial data sets (See Table 6) illustrates this argument. Since the training data was actually generated by a 12 state HMM and the HMM technique was trained with $\sigma = 12$; thus the HMM model effectively captures the normal sequences. The results of HMM for artificial data sets are therefore better than for public data sets, but still slightly worse than other techniques because of the poor performance of the $(1 + 1)$ FSA. When the normal sequences were generated using an HMM, the performance improves significantly. The hidden state sequences, obtained as a intermediate transformation of data, can actually be used as input data to any other technique discussed here. The performance of such an approach will be investigated as a future direction of research.

7 Reference Based Analysis Framework for Symbolic Sequences

The results on different data sets in Section 6 reveal that no one technique is clearly superior to others. Most techniques show consistency in performance on public data sets belonging to one domain but show different performance for data sets from a different domain. This indicates a relationship between the techniques and the nature of the data. In the artificial data sets generated from the data generator as well as the altered RVP data sets, we further studied this relationship by modifying the nature of the data using one or more tunable parameters. These observations motivate a deeper study of the relationship between a technique and a data set.

From this section onwards, we study the relationship between the anomaly detection techniques and the nature of data. Using the RBA framework [9], we characterize symbolic sequence data. We visualize the symbolic sequences using these characteristics which is useful to understand various aspects of the sequence data such as how different are the normal sequences from the anoma-

lous sequences and how similar are the normal sequences to each other. We then show how different anomaly detection techniques evaluated in Section 6 rely on one or more of such characteristics to detect anomalies. Using these characteristics, we propose two novel anomaly detection techniques for symbolic sequences, called WIN_{1D} and WIN_{2D}, which show consistently superior performance over the existing techniques across the different data sets.

8 Characterizing Sequence Data

The results on different data sets in Section 6 reveal that no one technique is clearly superior to others. Most techniques show consistency in performance on public data sets belonging to one domain but show different performance for data sets from a different domain. This indicates a relationship between the techniques and the nature of the data. In the artificial data sets generated from the data generator as well as the altered RVP data sets, we further studied this relationship by modifying the nature of the data using one or more tunable parameters. These observations motivate a deeper study of the relationship between a technique and a data set. To facilitate such a study, we first characterize a data set and then identify the relationship between a technique and the data characteristics.

We characterize a test sequence data set containing of normal and anomalous sequences with respect to a base data set, containing only normal sequences[4]. We describe different alternatives to characterize sequence data which are used by one or more of the techniques discussed in this paper, as will be discussed in Sections 9 and 10.

8.1 1-D Frequency Profiles

The first characterization is motivated from window based techniques that rely on the frequency of a k length window in a given sequence for anomaly detection. In this section we refer to a k length window as a k-window for brevity. Each k-window is associated with a frequency (denoted as f_k), i.e., the number of times it occurs in the training sequences.

A *1-D frequency profile* for a test sequence can be constructed as follows. First, all k-windows from the test sequence are extracted and their frequencies f_k are computed from the training sequences. The frequencies are "binned" into a fixed number (p) of *bins*. Since windows with $f_k = 0$ are of special interest, the first bin stores the windows with exactly $f_k = 0$. The other $p - 1$ bins divide the range $1 : max$ into equal width intervals, where max is the maximum frequency of any window in the given data set. The values in each bin are normalized to lie between 0 and 1 by dividing them by the total number

[4] This framework to characterize a given data set with respect to a base or reference data set was originally proposed for characterizing categorical data [9].

of windows in the given sequence. Thus each test sequence can be mapped into a $\Re^p$ space.

8.1.1 Average 1-D Frequency Profiles

To characterize a given test data set, we aggregate the 1-D frequency profiles. We construct the *average 1-D frequency profiles* for the normal test sequences and anomalous test sequences separately. It should be noted that the average profile might not be the best representation of the profiles. For example, let the test set contain 4 anomalous sequences. Using four bins ($p = 4$), let the frequency profiles for the four anomalous sequences be $(1.00, 0, 0, 0)$, $(0, 1.00, 0, 0)$, $(0, 0, 1.00, 0)$, and $(0, 0, 0, 1.00)$. The average frequency profile for the anomalous sequences will be $(0.25, 0.25, 0.25, 0.25)$ which does not provide an accurate representation of the actual profiles. But if the individual frequency profiles are similar to each other, the average profile will be representative.

A test sequence data set can be characterized with respect to a normal data set by taking the difference between the average 1-D frequency profiles for normal and anomalous test sequences. We will describe how this characteristic can be used to explain the behavior of window based techniques in Section 9.

8.2 2-D Frequency Profiles

The second characterization is motivated from Markovian techniques that rely on the frequency of a k length window as well as the frequency of the $k-1$ length prefix of the window, in a given sequence for anomaly detection. Thus, each k-window is associated with a tuple (f_k, f_{k-1}), where f_k is the frequency of occurrence of the k-window and f_{k-1} is the frequency of occurrence of the $k-1$ length prefix of the given k-window in the training sequences.

A *2-D frequency profile* for a test sequence can be constructed as follows. First, all k-windows from the test sequence are extracted and the associated tuples (f_k, f_{k-1}) are computed from the training sequences. The f_k frequencies are binned into p bins in the same manner as the 1-D frequency profiles. Similarly, the f_{k-1} frequencies are binned into p bins. Thus, every tuple (f_k, f_{k-1}) is assigned to a "cell" (or grid) on a $p \times p$ grid. The values in each cell are normalized to lie between 0 and 1 by dividing them by the total number of windows in the given sequence. Thus each test sequence can be mapped into a $\Re^{p \times p}$ space.

Note that the column aggregation of the *2-D frequency profile* for a test sequence will give the *1-D frequency profile* for the given test sequence.

8.2.1 Average 2-D Frequency Profiles

To characterize a given sequence data set, using the 2-D frequency profiles, we follow the same procedure as for 1-D frequency profiles. The frequency profiles for normal and anomalous sequences are aggregated separately to obtain an

average normal 2-D frequency profile and an average anomalous 2-D frequency profile, respectively.

A test sequence data set can be characterized with respect to a normal data set by taking the difference between the average 2-D frequency profiles for normal and anomalous test sequences. We will describe how this characteristic can be used to explain the behavior of Markovian techniques in Section 9.

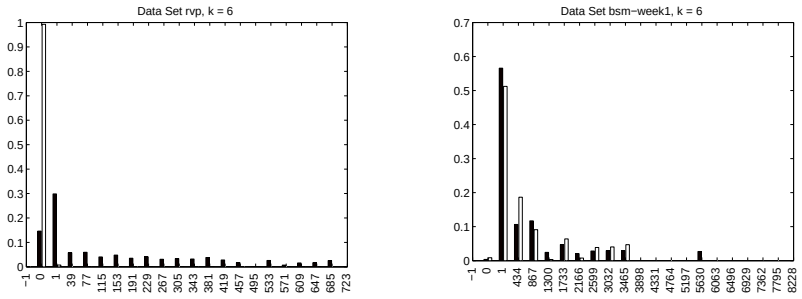
9 Relationship Between Performance of Techniques and Frequency Profiles

In this section we relate the performance of the window based ($tSTIDE$) and Markovian techniques (FSA, $FSAz$, PST, and $RIPPER$) to the 1-D and 2-D frequency profiles defined in Section 8.1.

9.1 $tSTIDE$

The performance of $tSTIDE$ can be explained using the 1-D frequency profiles described in the previous section. The anomaly score assigned by $tSTIDE$ is inversely proportional to the frequency of the k-windows in a given sequence. Hence the difference in the 1-D frequency profiles for normal and anomalous test sequences determines the relative performance of $tSTIDE$ on a given test data set.

For example, the average 1-D frequency profiles for rvp data set in Figure 4(a) are significantly different, and hence the performance of $tSTIDE$ is 90% (See Table 4). For bsm-week1 data set in Figure 4(b), the difference is not significant, and hence the performance of $tSTIDE$ is relatively poor (=20%).



(a) rvp (b) bsm-week1

Fig. 4 Average 1-D frequency profiles for 6-windows.

9.2 FSA

The $tSTIDE$ technique distinguishes between normal and anomalous test sequences in terms of the frequency of the k-windows, f_k. Often, f_k alone is not distinguishing enough (see Figure 4(b)). The FSA technique addresses this issue by considering the frequency of a k-window as well as the frequency of the $k-1$ length suffix of the k-window.

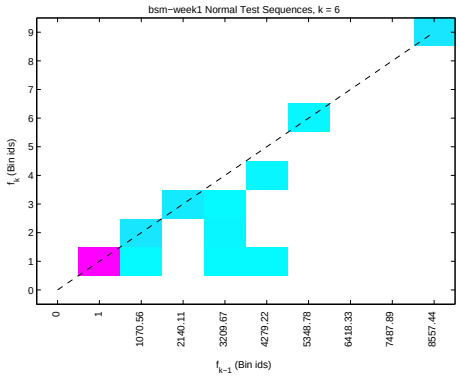
The performance of FSA can be explained using the 2-D frequency profiles described in previous section. FSA assigns anomaly score to a sequence using the values f_k and f_{k-1} for every k window. Hence the difference in the average 2-D frequency profiles for normal and anomalous sequences determines it relative performance on the given data set.

For example, the average 2D frequency profiles for the bsm-week1 data set are shown in Figures 5(a) and 5(b) for normal and anomalous sequences, respectively. The color of each cell represents the magnitude of the relative proportion of k-windows falling in that cell. We compare the two profiles with the 1D frequency profiles shown in Figure 4(b). The absolute difference between normal and anomalous frequency profiles is shown in Figure 5(c) with marker "+" indicating that normal test sequences had higher value for that cell than the anomalous test sequences, and marker "△" indicating that normal test sequences had lower value for that cell than the anomalous test sequences. Figures 6 and 7 show the plots (differences only) for other public data sets. Note that if the 2D profiles are collapsed onto the y-axis, we will get the corresponding 1D profiles. We note that even though the normal and anomalous sequences are not differentiable when only f_k is considered, the difference is significant when both f_k and f_{k-1} are considered. This is the reason why FSA performs better than $tSTIDE$ on the bsm-week1 data set.

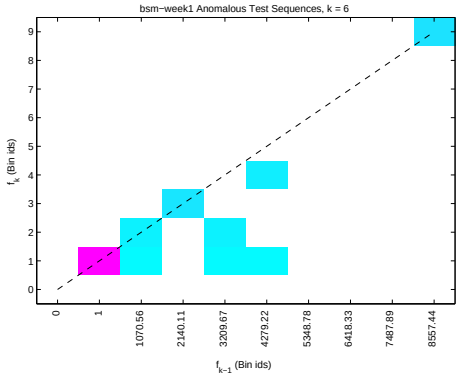
*Comparing $tSTIDE$ and FSA*The key distinction between $tSTIDE$ and FSA is that the former technique makes use of the frequencies of k-windows while the latter makes use of the frequencies of k-windows and the frequencies of their $k-$ length suffixes. This distinction is illustrated in Figure 8 which shows the scores assigned by $tSTIDE$ and FSA to windows, $w(f_k, f_{k-1})$. These scores are also referred to as likelihood scores and are the inverse of the anomaly score of the windows. Since $f_k \leq f_{k-1}$, the entries above the lower diagonal are ignored.

FSA ignores the k-windows for which $f_{k-1} = 0$, i.e., the bottom left corner of Figure 8(b). It is clear that the scores assigned by $tSTIDE$ are independent of f_{k-1} and are linearly proportional to f_k. Thus only high frequency windows will be assigned a high likelihood score by $tSTIDE$. But for FSA, k-windows with low f_k can still be assigned a high score, if the corresponding value of f_{k-1} is also low. This key difference accounts for a key strength and weakness of $tSTIDE$ and FSA.

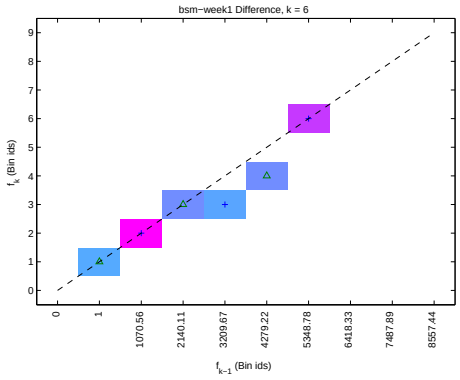
Consider a scenario in which the training data set is not pure but contains one anomalous sequence, such that most of the k-windows (for a given value of k) do not occur in any other training sequence. Let there be a truly anomalous



(a) Normal Test Sequences.

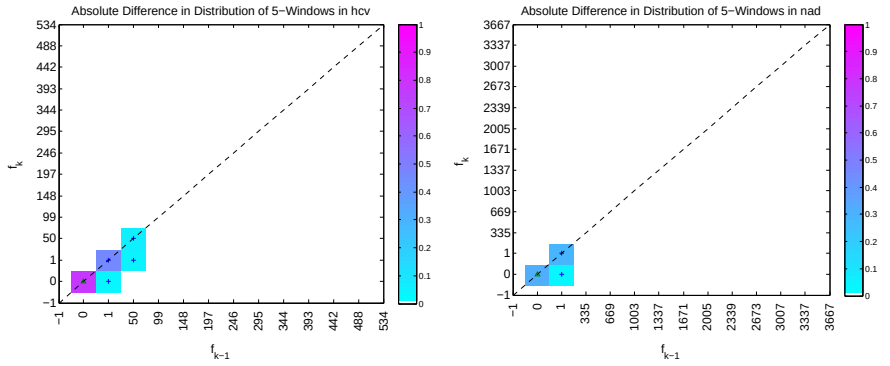


(b) Anomalous Test Sequences.



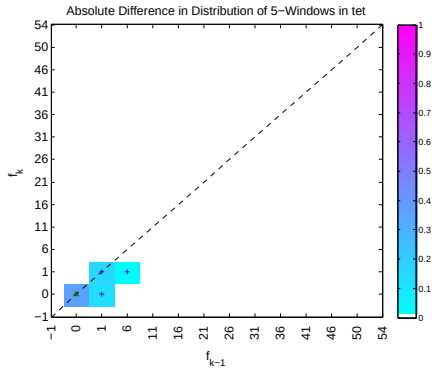
(c) Difference.

Fig. 5 2D Average frequency profiles for bsm-week1 data set ($k = 6$).

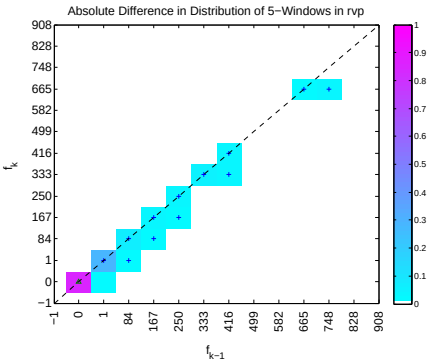


(a) hcv ($\Delta = 1.56$)

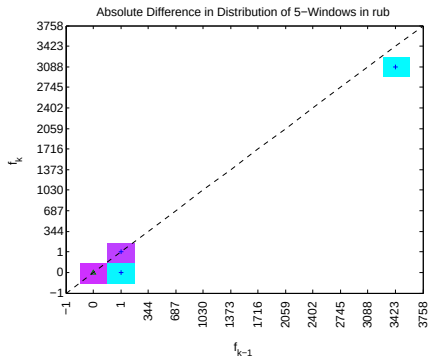
(b) nad ($\Delta = 0.64$)



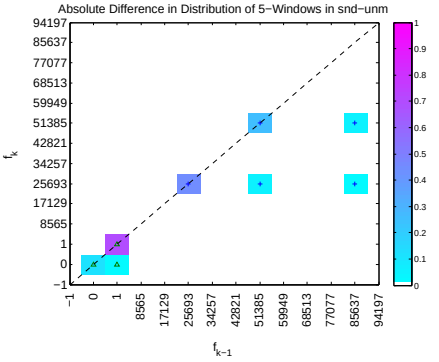
(c) tet ($\Delta = 0.73$)



(d) rvp ($\Delta = 1.71$)



(e) rub ($\Delta = 1.62$)



(f) snd-unm ($\Delta = 1.68$)

Fig. 6 Absolute difference in 2D frequency profiles for public data sets for $k = 6$ (hcv – snd-unm).

sequence in the test data set which is similar to the one anomalous training sequence. Most of the k-windows extracted from this test sequence will have $f_k = 1$. $tSTIDE$ will assign a high anomaly score to this test sequence. Though the value of f_{k-1} cannot be guaranteed, it is likely that $f_{k-1} \approx 1$. Thus FSA will assign a high likelihood score to the k-windows of the anomalous test sequence, and hence assign it a low anomaly score. Thus $tSTIDE$ is a better technique in this scenario.

Now consider a different scenario, in which the training data set contains a sequence that consists of k-windows that do not occur in any other training sequence, but are normal. Let the test data contain one truly normal sequence similar to this training sequence. $tSTIDE$ will assign a high anomaly score to this test sequence because the windows extracted from this sequence will have $f_k = 1$. But, similar to the argument for the previous scenario, FSA will assign a low anomaly score. Thus FSA is a better technique in this scenario.

To summarize, $tSTIDE$ is more robust when the training data might not be pure, i.e., it might contain anomalous sequences. $FSAz$ is a better choice when the training data has rare but normal patterns (windows) that have to be learnt.

9.3 $FSAz$

One issue with FSA is that it ignores the k-windows for which $f_k = f_{k-1} = 0$. But often, such windows can differentiate between normal and anomalous sequences. The plots of differences between the 2D average frequency profiles for normal and anomalous sequences, shown in Figures 6 and 7, show that for several data sets, anomalous test sequences have a higher proportion of such windows than the normal data sets. Our proposed technique, $FSAz$, utilizes this information by assigning a likelihood score of 0 to such sequences, instead of ignoring them. This makes $FSAz$ perform better than FSA for most data sets.

9.4 PST

An issue with FSA (and $FSAz$), as noted earlier, is that they estimate the conditional probability of a symbol, based on its fixed length history, even if the history occurs once in the training sequences. Thus such estimates can be unreliable, and hence make the techniques highly susceptible to presence of anomalies in the training set. PST addresses this issue by conditioning the probability of a symbol on its k length history, only if the history occurs a significant number of times in the training sequences. If the frequency of the history is low, i.e., the conditional probability estimate is unreliable, it uses the longest suffix of the history which satisfies the reliability threshold.

The score assigned by PST to a k-window is lower bounded by the score assigned by $FSAz$. The actual score assigned by PST not only depends on f_k,

f_{k-1}, but also on δ, which is a threshold on f_{k-1}. The value of δ is determined using user-defined parameters and the training sequences (See Section 3.3.2). For a given k-window, if $f_{k-1} \geq \delta$ the score assigned by PST is same as $FSAz$. Otherwise, PST chooses the longest suffix of the k window of length j ($2 \leq j \leq k$), such that $f_{j-1} \geq \delta$. If $f_1 < \delta$, PST assigns the score equal to the probability of observing the last (k^{th}) symbol of the given window.

For example, Figures 9(a)–9(e) show the difference in the frequency profiles for normal and anomalous test sequences for the rub data set for different values of k. To assign a score to a k-window for $k = 6$, the PST technique will first consider the frequency profile for $k = 6$. Let us assume that the k-window to be scored has $f_{k-1} < \lambda$. In this case PST will substitute the score with the score of a window of length $k - 1$ in the frequency profile for $k - 1$ length windows. If for the $k - 1$ window, $f_{k-2} \geq \lambda$, the corresponding score will be used, otherwise the frequency profile for $k - 3$ is considered, and so on.

Using this understanding of PST, we can explain why PST performs significantly poorly than $FSAz$ for most of the public data sets. Let us consider the data set rub. Figure 9(a) shows difference in the frequency profiles of normal and anomalous test sequences for $k = 6$. The distinguishing cells in the profile are mostly located in the bottom left corner, and there is a single distinguishing cell in the upper right corner. Both PST and $FSAz$ will assign similar scores to the k-windows belonging to the cell in the upper right corner. For the cell in the bottom leftmost corner, $FSAz$ will assign a 0 score, and for other cells $FSAz$ will assign a higher score. Thus $FSAz$ will be able to distinguish between normal and anomalous test sequences, which supports our experimental finding that the performance of $FSAz$ on this data set is (0.88). For PST, all k-windows belonging to the cells in the bottom left corner will have $f_{k-1} < \lambda$, and hence will be substituted with scores for shorter suffix of the k-windows. Thus the scores for windows to the bottom leftmost cell in Figure 9(a) will be scored same as the shorter windows (of length $j < k$) in Figures 9(b)–9(e) for which $f_{j-1} \geq \delta$. But it is evident from the plots that normal and anomalous test sequences are not significantly distinguishable for higher values of f_{j-1}. This is reason why PST performs poorly for this data set (0.28).

While the above mentioned behavior of PST is an obvious disadvantage for most of the data sets, it can also favor PST in certain cases. For example, PST performs well in comparison to $FSAz$ on the artificial data set $d6$. The frequency profiles of normal and anomalous test sequences for $d6$ are shown for different values of k in Figures 10(a) – 10(e). We observe that the frequency profiles for normal and anomalous test sequences are not distinguishable for $k = 6$ and hence $FSAz$ performs poorly (0.38). But when frequency profiles for lower values of $j \leq 3$ are considered by the PST, the profiles for normal and anomalous sequences are relatively more distinguishable (even for larger values of f_{j-1}) and hence PST performs better (0.68).

9.5 $RIPPER$

The motivation behind $RIPPER$ is same as PST, i.e., if the fixed length history of a symbol in a test sequence does not have a reliable frequency in the training sequences, the symbol is conditioned on a subset of the history. The difference being that the subset is not the suffix of the history (as is the case with PST), but a subsequence of the history.

$RIPPER$, like PST, assigns score to a k-window which is lower bounded by the score assigned by $FSAz$. The actual score assigned by PST depends on the $RIPPER$ rule that is "fired" for the $k-1$ prefix of the given window. If the target of the fired rule matches the k^{th} symbol of the given window, the likelihood score is 1, else the likelihood score is the inverse of the confidence associated with the rule. It is difficult to analytically estimate the actual scores assigned by $RIPPER$, but generally, the scores assigned by $RIPPER$ are higher than $FSAz$ but lower than PST.

As mentioned earlier, the scores assigned by $RIPPER$ are same as $FSAz$ for higher values of f_k. For lower values of f_k the scores depend on the distribution of k-windows in the training data set as well as how the underlying classifier ($RIPPER$) learns the rules and what is the order in which the rules are applied. Generally speaking, it can be stated that the scores assigned by $RIPPER$ to such windows is greater than 0 but lower than the score assigned by PST to such windows.

The above mentioned behavior of $RIPPER$ results in its poor performance in cases in which the cells with lower values of f_k are distinguishing and the anomalous test sequences have higher proportion of windows in that cell than the normal test sequences. $RIPPER$ assigns a higher overall likelihood score to the anomalous test sequences and hence is not able to distinguish them from normal sequences. For all PFAM data sets the distinguishing cells have lower f_k value, resulting in poor performance of $RIPPER$. For UNM data sets, the distinguishing cells have higher values for f_k and hence the performance of $RIPPER$ is very close to that of $FSAz$.

10 Impact of Nature of Similarity Measure on Performance of Anomaly Detection Techniques

Kernel based techniques (kNN and CLUSTER) are distinct from the window based and Markovian techniques because they rely on the similarity between a test sequence and training sequences to assign anomaly score to the test sequence. Thus their performance can be explained using the *average similarity to training sequences characteristic*, as described in Section 3.1.

One distinction between normal and anomalous sequences is that normal test sequences are expected to be more similar (using a certain similarity measure) to training sequences, than anomalous test sequences. If the difference in similarity is not large, this characteristic will not be able to accurately distinguish between normal and anomalous sequences. This characteristic is utilized

by kernel based techniques (kNN and CLUSTER) to distinguish between normal and anomalous sequences.

For example, Figure 11(a) shows the histogram of the average ($nLCS$) similarities of test sequences in the artificial data set $d1$ to the training sequences. The normal test sequences are more similar to the training sequences, than the anomalous test sequence. This indicates that techniques that use similarity between sequences to distinguish between anomalous and normal sequences will perform well for this data set. From Table 6, we can observe that the performance of CLUSTER as well as kNN is 100% on $d1$. A similar histogram for data set $d6$ is shown in Figure 11(b), which shows that average similarities of normal test sequences and the average similarities of anomalous test sequences are very close to each other. This confirms the observation in Table 6 that CLUSTER and kNN should perform poorly for this data set.

We quantify the above characteristic by computing the average sequence similarity for each test sequence. Let the average of the average similarities for normal test sequences be denoted as s_n, and average of the average similarities for anomalous test sequences be denoted as s_a. If for a given data set, the difference $s_n - s_a$ is large, kNN and CLUSTER are expected to perform well on that data set, and vice-versa.

	hcv	nad	tet	rvp	rub	snd-unm	snd-cert	bsm-week1	bsm-week2	bsm-week3
s_n	0.53	0.48	0.67	0.82	0.75	0.99	0.99	0.97	0.98	0.97
s_a	0.38	0.38	0.37	0.36	0.37	0.50	0.38	0.88	0.81	0.73
$s_n - s_a$	0.15	0.10	0.30	0.46	0.38	0.49	0.61	0.09	0.17	0.24

Table 8 Values of s_n, s_a for the public data sets.

	d1	d2	d3	d4	d5	d6
s_n	0.87	0.87	0.86	0.86	0.86	0.86
s_a	0.45	0.63	0.63	0.73	0.76	0.78
$s_n - s_a$	0.42	0.24	0.23	0.13	0.10	0.08

Table 9 Values of s_n, s_a for the artificial data sets.

Tables 8 and 9 show the values of s_n, s_a, and $s_n - s_a$, for the real and artificial data sets, respectively. The performance of both kNN and CLUSTER is highly correlated with the difference $s_n - s_a$.

11 Using RBA Features for Anomaly Detection

A key aspect of the RBA framework is that it maps data instances into a multivariate continuous space, where normal and anomalous instances can be distinguished from each other. Thus, applying the RBA framework is equivalent to extracting features from the sequence data set. In this section, we

propose two novel techniques based on these features to detect anomalies in a given test data set. We denote the novel techniques as WIN_{1D} and WIN_{2D}, since they utilize the 1-D and 2-D frequency profiles discussed in Sections 8.1 and 8.2, respectively.

The motivation behind these two techniques is the fact that several existing techniques implicitly utilize the difference between the relative frequencies of the k-windows to distinguish between normal and anomalous sequences (See Section 9).

The algorithm for the first technique, WIN_{1D}, is as follows:

$WIN_{1D}(k, p, nn, \mathbf{S}, \mathbf{T})$

1. For each training sequence $T_j \in \mathbf{T}$, calculate its 1-D frequency profile with respect to $\mathbf{T}$ (denoted as $\dot{T}_j$) with window size k and number of bins as p.
2. For each test sequence $S_i \in \mathbf{S}$, calculate its 1-D frequency profile with respect to $\mathbf{T}$ (denoted as $\dot{S}_i$) with window size k and number of bins as p.
3. For each "mapped" test sequence, $\dot{S}_i$, calculate its anomaly score as equal to the distance to its nn^{th} nearest neighbor in $\dot{\mathbf{T}}$[5] using *Euclidean* distance metric.

The algorithm for the second technique, WIN_{2D}, is as follows:

$WIN_{2D}(k, p, nn, \mathbf{S}, \mathbf{T})$

1. For each training sequence $T_j \in \mathbf{T}$, calculate its 2-D frequency profile with respect to $\mathbf{T}$ (denoted as $\ddot{T}_j$) with window size k and number of bins as p.
2. For each test sequence $S_i \in \mathbf{S}$, calculate its 2-D frequency profile with respect to $\mathbf{T}$ (denoted as $\ddot{S}_i$) with window size k and number of bins as p.
3. For each "mapped" test sequence, $\ddot{S}_i$, calculate its anomaly score as equal to the distance to its nn^{th} nearest neighbor in $\ddot{\mathbf{T}}$[6]using *Euclidean* distance metric.

The inputs to both techniques are the training data set, $\mathbf{T}$, test data set, $\mathbf{S}$, window size, $k(\geq 2)$, number of bins, $p(\geq 2)$, and number of nearest neighbors to analyze, nn.

11.1 Results on Public and Artificial Data Sets

We evaluate the performance of the proposed techniques on the public and artificial data sets described in Section 4.

[5] $\dot{\mathbf{T}}$ is the set of "mapped" training sequences using the 1-D frequency profiles.
[6] $\ddot{\mathbf{T}}$ is the set of "mapped" training sequences using the 2-D frequency profiles.

Sensitivity to Parameters

We first investigate the sensitivity of the different parameters on the performance of WIN_{1D} and WIN_{2D}. The techniques gave best overall performance for window size $k = 6$, which was also the best performing window size for the existing window based and Markovian techniques. The performance of both techniques was not sensitive to the number of nearest neighbors. The techniques gave best performance when the number of bins used to construct the profile, p, was low (≈ 3). For larger values of p, the dimensionality of the mapped data increased, and hence the performance of the distance based anomaly detection technique deteriorated.

For the results provided in subsequent section, the optimal parameter settings were found by testing on a validation set for different combinations of the parameters (p, k, nn) and using the combination the provides best average results across all data sets. The results are shown for $p = 5$, $k = 6$, and $nn = 5$.

Comparison with All Existing Techniques

The first set of results show how WIN_{1D} and WIN_{2D} compare against the state of art techniques, discussed in Section 3, on the different public and artificial data sets. The comparison of average performance of the proposed techniques with the existing techniques is shown in Figure 12. Notably, WIN_{1D}, which is based on $tSTIDE$, shows better performance than $tSTIDE$, and WIN_{2D}, which is based on $FSAz$, shows better performance than $FSAz$ on both public and artificial data set. Overall, WIN_{2D} performs significantly better than all existing techniques on average across all public and artificial data sets.

The reason the proposed techniques perform better than the existing techniques is because of the way the windows are utilized by the proposed techniques. For example, let us consider $tSTIDE$ and WIN_{1D}. Both of these techniques use the frequency of k-windows to distinguish between the normal and anomalous test sequences. But $tSTIDE$ weights the windows in a test sequence by their frequencies and the sums the total weights to get an inverse of the anomaly score. On the other hand, WIN_{1D} bins the windows based on their frequency, and then uses the normalized bin counts as features. By using a nearest neighbor approach, WIN_{1D} "learns" weights on different windows to achieve best separability between the normal and anomalous test sequence. Same holds true for $FSAz$ and WIN_{2D}.

Comparison with Best Existing Technique

The strength of the RBA based techniques, WIN_{1D} and WIN_{2D}, is that they distinguish between normal and anomalous test sequences in a multi-dimensional space, while most of the existing techniques operate along one or a limited subset of the dimensions. In the second set of results we assess

if this strength allows the RBA based techniques to outperform the existing techniques.

In Table 10, we compare the accuracy results for WIN_{1D} and WIN_{2D} against the best existing technique for each public data set. The results show that both WIN_{1D} and WIN_{2D} are strictly better or comparable with the best existing technique for almost all of the public data sets. The same inference can be drawn from the AUC results for the public data sets in Table 11.

For artificial data sets, the performance of WIN_{2D} is still significantly better than the best existing technique for each data set as shown in Tables .12 and 13. Notably, for the artificial data sets, the performance of WIN_{1D} is relatively worse than the best existing technique.

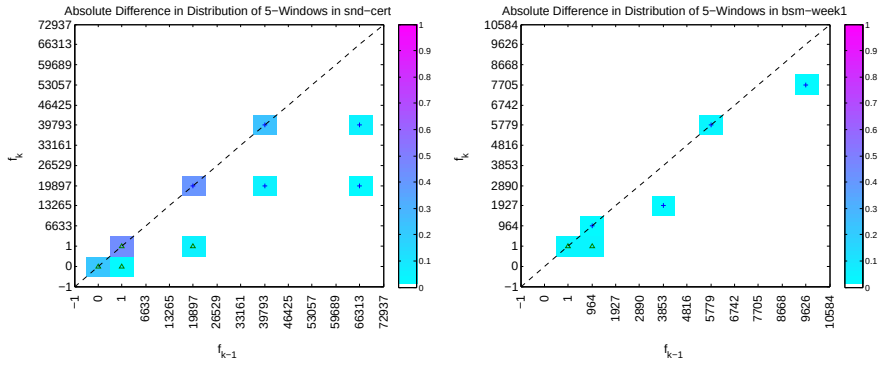
For artificial data sets, PST was found to be the best technique while both FSA and $FSAz$ were found to perform poorly for many artificial data sets. The reason was that the artificial data sets were designed to break FSA and $FSAz$, while PST, which utilizes the frequencies of varying length suffixes of the k length windows, was able to distinguish between the normal and anomalous test sequences. By using WIN_{2D}, the behavior of PST is captured and improved, and hence WIN_{2D} outperforms PST on the artificial data sets.

12 Conclusions and Future Work

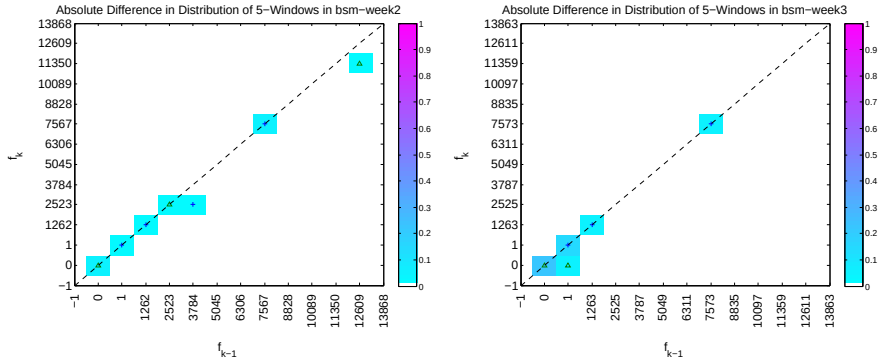
In this paper we have shown how the RBA framework can be used in the context of anomaly detection for symbolic sequences. Visualizing symbolic sequences is challenging, especially when the sequences are of varying length. Using the RBA framework we provide a visualization scheme for symbolic sequences.

The RBA based mapping for the symbolic sequences is motivated from the existing techniques that use fixed length windows as a unit of analysis. In this chapter we have shown how, using the RBA based features, one can understand the performance of the different existing techniques. Moreover, the framework can also be used to identify the fundamental differences between techniques. For example, $tSTIDE$ and FSA are shown to be highly different from each other since they handle k-windows in distinct manner. The framework also allows to identify the weaknesses of each technique. For example, the poor performance of PST on most of the real data sets could be explained using the framework. The same framework can also be used to construct scenarios in which a given technique would perform well or poorly.

The analysis of the various distinguishing characteristics can also aid in choosing optimal values of parameters for different techniques. For example, Figure 9 shows the magnitude of difference between normal and anomalous sequences in rub data set for different values of window size k. The maximum difference occurs when $k = 5$ or 6. Our results indicate that all techniques that depend on window size as a parameter give optimal performance for these values of k. Similarly, for kNN and $CLUSTER$, the difference in the corresponding characteristic for normal and anomalous test sequence, can be



(a) snd-cert ($\Delta = 1.54$) (b) bsm-week1 ($\Delta = 0.23$)



(c) bsm-week2 ($\Delta = 0.27$) (d) bsm-week3 ($\Delta = 0.61$)

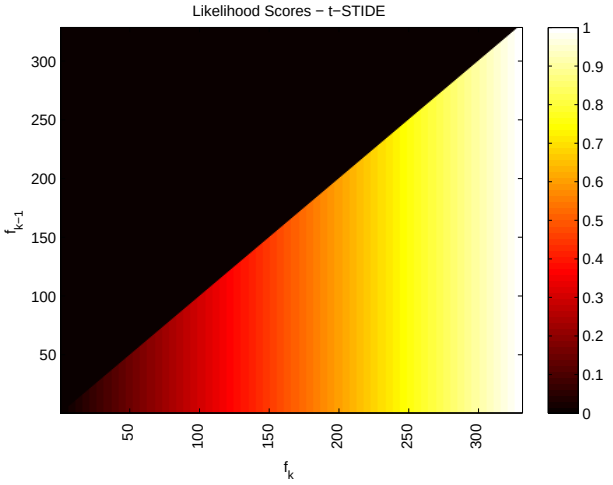
Fig. 7 Absolute difference in 2D frequency profiles for public data sets for $k = 6$ (snd-cert – bsm-week3).

	PFAM					UNM		DARPA			
	hcv	nad	tet	rvp	rub	snd-unm	snd-cert	bsm-week1	bsm-week2	bsm-week3	**Avg**
WIN_{1D}	0.92	0.74	0.52	0.90	0.88	0.82	0.88	0.30	0.60	0.66	0.72
WIN_{2D}	0.92	0.76	0.82	0.92	0.92	0.84	0.88	0.50	0.60	0.66	0.78
Existing Best	0.92	0.74	0.50	0.90	0.88	0.82	0.88	0.50	0.56	0.66	0.74

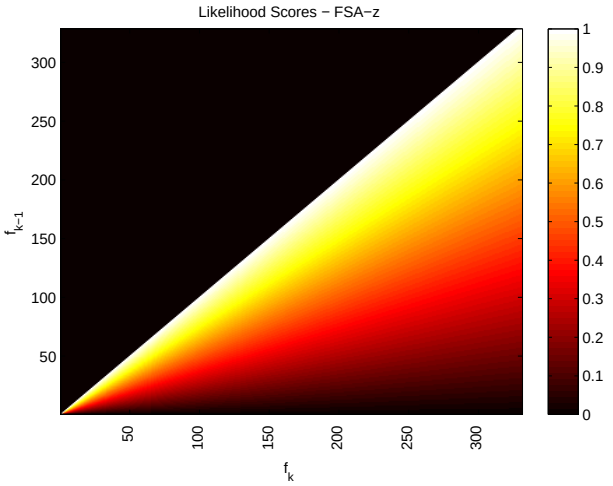
Table 10 Comparing accuracy of WIN_{1D} and WIN_{2D} against best existing technique for public data sets.

	PFAM					UNM		DARPA			
	hcv	nad	tet	rvp	rub	snd-unm	snd-cert	bsm-week1	bsm-week2	bsm-week3	**Avg**
WIN_{1D}	1.00	0.98	0.98	1.00	1.00	0.99	0.98	0.75	0.92	0.92	0.95
WIN_{2D}	1.00	0.99	0.99	1.00	1.00	0.99	0.98	0.91	0.93	0.92	0.97
Existing Best	1.00	0.98	0.98	1.00	1.00	0.99	0.96	0.88	0.91	0.97	0.97

Table 11 Comparing AUC of WIN_{1D} and WIN_{2D} against best existing technique for public data sets.



(a) $tSTIDE$.

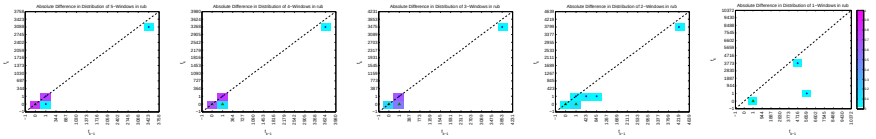


(b) $FSAz$

Fig. 8 Likelihood scores $L(f_k, f_{k-1})$, assigned by different techniques.

	d1	d2	d3	d4	d5	d6	**Avg**
WIN_{1D}	1.00	0.92	0.58	0.52	0.34	0.64	0.67
WIN_{2D}	1.00	0.96	0.81	0.76	0.71	0.74	0.83
Existing Best	1.00	0.84	0.82	0.76	0.68	0.68	0.80

Table 12 Comparing accuracy of WIN_{1D} and WIN_{2D} against best existing technique for artificial data sets.



(a) $k = 6$ ($\Delta = 1.62$) (b) $k = 5$ ($\Delta = 1.62$) (c) $k = 4$ ($\Delta = 1.37$) (d) $k = 3$ ($\Delta = 0.32$) (e) $k = 2$ ($\Delta = 0.13$)

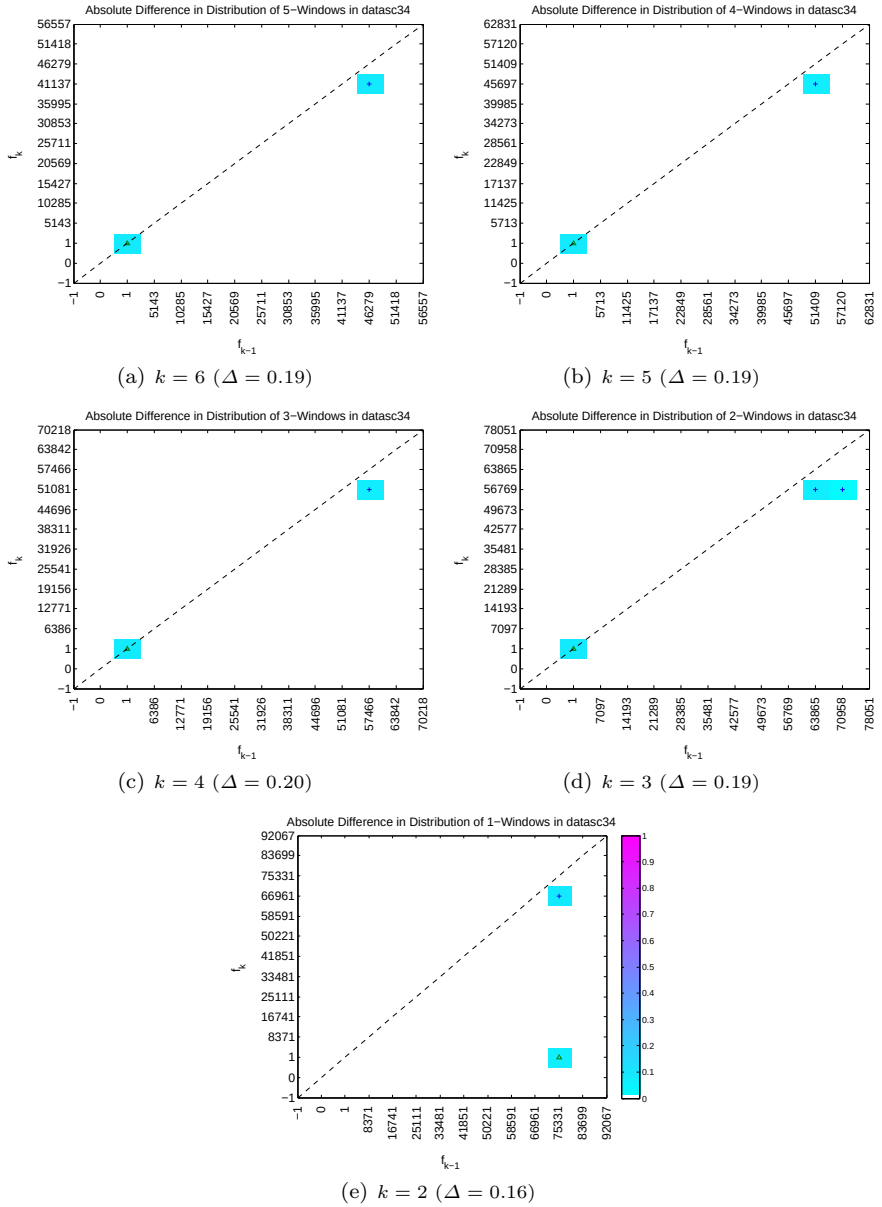
Fig. 9 Absolute difference in frequency profiles for rub data set.

calculated for different values of the parameter k. The value of k that results in maximum difference in terms of the characteristic, is likely to give best performance on that data set. One could argue that given a labeled validation data set, a technique can be evaluated for different parameter values to obtain the optimal value. But using the proposed framework, the analysis needs to be done only for a characteristic, without having to test every technique that depends on that characteristic.

The most significant outcome of applying the RBA framework to symbolic sequences is that the features obtained from the mapping can be used to develop powerful anomaly detection techniques, which outperform the existing techniques. Moreover, the RBA based techniques are shown to better than the best existing technique for most of the data sets. Thus instead of using different existing techniques which are optimal for different data sets, RBA provides one best technique across a variety of data sets. This is a significant step towards the ultimate goal for the anomaly detection research, which is to find a technique that can perform well across all application domains.

References

1. V. Chandola, A. Banerjee, V. Kumar, ACM Computing Surveys **41**(3) (2009)
2. V. Hodge, J. Austin, Artificial Intelligence Review **22**(2), 85 (2004). DOI http://dx.doi.org/10.1023/B:AIRE.0000045502.10941.a9
3. A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, J. Srivastava, in *Proceedings of SIAM International Conference on Data Mining* (SIAM, 2003)
4. P. Sun, S. Chawla, B. Arunasalam, in *In SIAM International Conference on Data Mining* (2006)
5. S. Forrest, S.A. Hofmeyr, A. Somayaji, T.A. Longstaff, in *Proceedinges of the ISRSP96* (1996), pp. 120–128. URL citeseer.ist.psu.edu/forrest96sense.html
6. S.A. Hofmeyr, S. Forrest, A. Somayaji, Journal of Computer Security **6**(3), 151 (1998). URL citeseer.ist.psu.edu/hofmeyr98intrusion.html
7. S. Forrest, C. Warrender, B. Pearlmutter, in *Proceedings of the 1999 IEEE ISRSP* (IEEE Computer Society, Washington, DC, USA, 1999), pp. 133–145
8. V. Chandola, V. Mithal, V. Kumar, in *Proceedings of International Conference on Data Mining* (2008)
9. V. Chandola, S. Boriah, V. Kumar, in *Proceedings of the ninth SIAM International Conference on Data Mining* (2009)
10. F.A. Gonzalez, D. Dasgupta, Genetic Programming and Evolvable Machines **4**(4), 383 (2003). DOI http://dx.doi.org/10.1023/A:1026195112518
11. B. Gao, H.Y. Ma, Y.H. Yang, in *Proceedings of International Conference on Machine Learning and Cybernetics* (IEEE, 2002), pp. 381–385



(a) $k = 6$ ($\Delta = 0.19$)

(b) $k = 5$ ($\Delta = 0.19$)

(c) $k = 4$ ($\Delta = 0.20$)

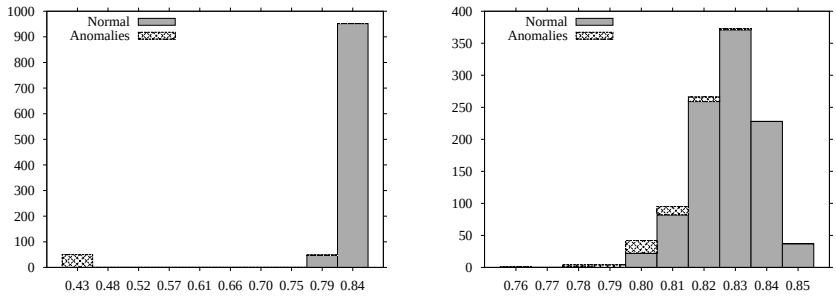
(d) $k = 3$ ($\Delta = 0.19$)

(e) $k = 2$ ($\Delta = 0.16$)

Fig. 10 Absolute difference in frequency profiles for $d6$ data set.

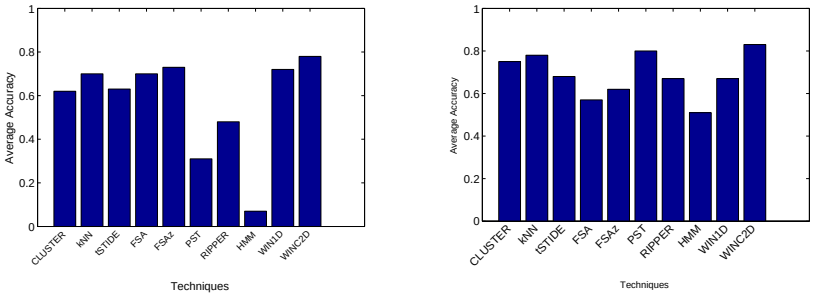
	d1	d2	d3	d4	d5	d6	**Avg**
WIN_{1D}	1.00	1.00	0.91	0.89	0.76	0.87	0.90
WIN_{2D}	1.00	1.00	0.98	0.98	0.96	0.98	0.98
Existing Best	1.00	0.96	0.98	0.98	0.96	0.95	0.97

Table 13 Comparing AUC of WIN_{1D} and WIN_{2D} against best existing technique for artificial data sets.



(a) Data Set $d1$. (b) Data Set $d6$.

Fig. 11 Histogram of Average Similarities of Normal and Anomalous Test Sequences to Training Sequences.



(a) Public Data Sets (b) Artificial Data Sets

Fig. 12 Comparison of average accuracies for WIN_{1D} and WIN_{2D}, and existing anomaly detection techniques.

12. Y. Qiao, X.W. Xin, Y. Bin, S. Ge, Electronics Letters **38**(13), 663 (2002)
13. W. Lee, S. Stolfo, P. Chan, in *Proceedings of the AAAI 97 workshop on AI methods in Fraud and risk management* (1997)
14. W. Lee, S. Stolfo, in *Proceedings of the 7th USENIX Security Symposium* (San Antonio, TX, 1998)
15. C.C. Michael, A. Ghosh, in *Proceedings of the 16th Annual Computer Security Applications Conference* (IEEE Computer Society, 2000), p. 21
16. E. Eskin, W. Lee, S. Stolfo, in *Proceedings of DISCEX* (2001). URL citeseer.ist.psu.edu/portnoy01intrusion.html
17. S. Budalakoti, A. Srivastava, M. Otey, in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, vol. 37 (2007), vol. 37
18. A.N. Srivastava, in *Proceedings of 2005 Joint Army Navy NASA Airforce Conference on Propulsion* (2005)
19. V. Chandola, S. Boriah, V. Kumar, in *CSIIRW '10: Proceedings of the 6th Annual Workshop on Cyber Security and Information Intelligence Research* (ACM, New York, NY, USA, 2010)
20. V. Chandola, A. Banerjee, V. Kumar, Anomaly detection for discrete sequences: A survey (2009). Submitted to TKDE
21. P.N. Tan, M. Steinbach, V. Kumar, *Introduction to Data Mining* (Addison-Wesley, 2005)
22. C.S. Leslie, E. Eskin, W.S. Noble, in *Pacific Symposium on Biocomputing* (2002), pp. 566–575

23. N. Kumar, V.N. Lolla, E.J. Keogh, S. Lonardi, C.A. Ratanamahatana, in *SDM* (2005)
24. S. Ramaswamy, R. Rastogi, K. Shim, in *Proceedings of the ACM SIGMOD international conference on Management of data* (ACM, 2000), pp. 427–438. DOI http://doi.acm.org/10.1145/342009.335437
25. D. Ron, Y. Singer, N. Tishby, Machine Learning **25**(2-3), 117 (1996). DOI http://dx.doi.org/10.1007/BF00114008
26. W.W. Cohen, in *Proceedings of the 12th International Conference on Machine Learning*, ed. by A. Prieditis, S. Russell (Morgan Kaufmann, Tahoe City, CA, 1995), pp. 115–123
27. A. Bateman, E. Birney, R. Durbin, S.R. Eddy, K.L. Howe, E.L. Sonnhammer, Nucleic Acids Res. **28**, 263 (2000)
28. R.P. Lippmann, et al, in *DARPA Information Survivability Conference and Exposition (DISCEX) 2000*, vol. 2 (IEEE Computer Society Press, 2000), vol. 2, pp. 12–26
29. V. Chandola, V. Mithal, V. Kumar, Comparing anomaly detection techniques for sequence data. Tech. Rep. 08-021, University of Minnesota, Computer Science Department (2008)