Signal Processing-based Anomaly Detection Techniques: A Comparative Analysis

Joseph Ndong
Université Pierre et Marie Curie
LIP6-CNRS, France
Email: joseph.ndong@lip6.fr

Kavé Salamatian
Université de Savoie Chambery Annecy
LISTIC PolyTech, France
Email: kave.salamatian@univ–savoie.fr

***Abstract*—In this paper, we present an analysis for anomaly detection by comparing two well known approaches, namely the Principal Component Analysis (PCA) based and the Kalman filtering based signal processing techniques. The PCA-based approach is coupled with a Karuhen-Loeve expansion (KL) to achieve higher improvement in the detection performance; on the other hand, based on a Kalman filter, we built a new method by combining statistical methods such as: gaussian mixture and a hidden markov modellers, which allows us to obtain performances better than those obtained with the PCA-KL expansion method. For this newer method, our approach consists of not assuming anymore that the Kalman innovation process is gaussian and white. In place, we are assuming that the real distribution of the process is a mixture of normal distributions and that, there is time dependency in the innovation that we will capture by using a Hidden Markov Model. We therefore derive a new decision process and we show that this approach results in an considerable decrease of false alarm rates. We validate the two comparative approaches over several different realistic traces.**

***Index Terms*—Anomaly Detection, Monitoring System, Kalman filter, GMM, HMM**

I. INTRODUCTION

The literature of the recent years has used two fundamental classes of monitoring techniques, to implement anomaly detection techniques for networking applications: PCA-based and Kalman-filter based methods. An anomaly detector consists essentially of two components: (i) an entropy reduction component and (ii) a decision component applying statistical tests to a *decision variable* issued from the first step. The entropy reduction step is here, to simplify the second step. When using statistical signal processing based techniques, entropy reduction is obtained by a predictive model that uses a model of normal behavior to forecast the values of the parameters to monitor. The prediction error obtained after filtering out the normal behavior model prediction has a smaller entropy than the initial signal, resulting in a considerable entropy reduction. A decision variable is thereafter derived as a function of the prediction error and fed to the decision stage. In the decision stage a statistical test is applied to the decision variable and an anomaly is detected if this variable exceed a threshold.

In PCA-based method, the predictive model used in the entropy reduction step is built by using a projection in a *low* dimensional *orthogonal* sub-space, that minimizes the approximation error. This subspace is derived using Principal Component Analysis (PCA) or more precisely a Karhunen-Loeve Transform (KLT). The decision variable in PCA-based techniques is obtained as, a *square sum of the prediction errors* made by projecting the observed signal in the PCA defined subspace (see [1], [2] for a detailed description).

Kalman-filter based techniques first, calibrate a Maximum-Likelihood based model for normal behavior modeling for the entropy reduction step. Thereafter the decision variable is obtained as the *innovation process* at the **output** of a Kalman-filter, that filters the normal behavior component from network observations [3].

In the two above cited methods, under the conditions that the observed signals follow a jointly gaussian distribution, the decision variable is known to converge to a zero mean, gaussian and white (uncorrelated) signal . However, realistic network traffic is well known to **not** follow a gaussian distribution. Moreover, anomalies can break the ***stationarity*** of the decision signal when they happen generally. These last two problems result in divergence from the basic assumptions (gaussianity) under which classical anomaly detectors are built. Therefore, the error probabilities bounds predicted by assuming a gaussian distribution at the decision variable are not tight enough to be used for a robust anomaly detection test. Prior work [1], [2], [4] have shown that, even after careful calibration of the normal behavior model, the decision variables still exhibit non-gaussian and correlated behaviors. This last problem explains the high false alarm rate observed when using classical anomaly detection approaches.

The fundamental issue in the area of anomaly detection in networks is the false alarm rate *vs.* detection rate trade-off. This trade-off is represented by the ROC (Receiver Operating Characteristics) curve [4]. In operational settings, one would like to attain a high detection rate (larger than 90%) with the lowest false alarm probability. The main aim of this paper is twofold: to see between the two methods, PCA or Kalman filtering, which one is the most robust towards deviation from the gaussianity assumptions, and to improve the false alarm versus detection rate trade-off by making a more robust anomaly detection decision. This will allow us to compare the two above methods and to show that the method based on Kalman filtering performs best.

The contribution of this paper is a careful comparative analysis of the decision step in PCA and Kalman-filter based anomaly detectors. Based on this analysis, we are proposing one approach to improve the robustness of the decision step. This approach is based on accepting that the distribution of the decision variable at the output of the anomaly detector is not an uncorrelated gaussian process. We will rather assume that, the decision variable follows a distribution that can be modelled by a Gaussian Mixture Model (GMM) with a *small* number of components.

The residual temporal correlation in the decision variable is modelled by a Hidden Markov Model (HMM) defined on the sequence of component index of the GMM model. Since anomalies might be rare, we assume that they might happen in some (but not all) components of the GMM. This means that, by inferring the states of the HMM calibrated on the residual variable, one is able to decide if an anomaly has happened.

The above model is not the only one that is able to capture the deviation from gaussian hypothesis as well as the residual correlation in the decision variable. However, we will show in this paper that the crude and relative complexity of the model is powerful enough to result in a considerable decrease in false alarm rate for a given detection ratio.

In summary our proposed robust decision scheme differs from previous work in the field of network anomaly detection in at least two ways: i) we do not assume that the innovation process does strictly remain a zero mean gaussian process. Instead, we assume that the real distribution of the process is a mixture of gaussians, and residuals can be split in different mixture components (where anomalies might or might not happen), ii) we use hidden markov models over the different mixture components to capture residual time dependencies that can be relevant to anomaly detection. Therefore, anomaly decision is not done as usual, using a simple threshold based decision but rather in a two step approach: we first use a Viterbi algorithm to estimate the Maximum a posteriori (MAP) estimates of the state sequence of the HMM; clearly the Viterbi path will select some of the HMM states (i.e some of the GMM components, because each HMM state contains part of the GMM components) able to capture all the low and high variations in the decision variable; thereafter we only apply thresholding in these selected states.

The organization of this paper is as follows. Section 2 describes the monitoring system we used. Section 3 deals with the methodology we adopt in our anomaly detection scheme. In section 4, we detail our calibration method and we validate our approach by showing efficient results, which we compare with the results obtained by the KL-PCA method. Section 5 concludes our work and fix some ideas for future works.

II. Architecture of the Monitoring System

We are assuming a novel network monitoring system shown in Figure 1. The monitoring system contains three functional blocks: data collection (by a NOC-Network Operations Center), a data analysis block and a decision process phase. The data collection block is responsible for receiving the flow of microscopic measurement coming from network equipment in form of SNMP or Netflow/IPFIX flows. This flow is

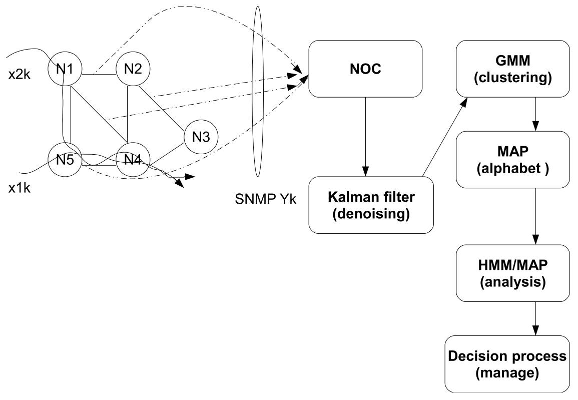

Fig. 1. A new monitoring system combining a Kalman filter for entropy reduction, a GMM for clustering, a HMM for time dependencies learning and finally the use of the Viterbi algorithm for decision manage.

aggregated and transformed to a vector of time sequences that are fed to other blocks. The Data analysis block calibrates the normal behavior model, by applying machine learning techniques like PCA or Maximum Likelihood analysis and generates the decision variable by filtering the expected normal behavior from the observations. The last block implements the decision process. Classically, the decision stage consists simply of a comparison with a threshold, *i.e.* if the decision variable exceeds the threshold, an anomaly is detected.

We propose in this paper a more elaborate decision scheme containing three new components shown in Fig. 1. These three new stages need to calibrate two models: a GMM to account for deviation from gaussian distribution of the decision variable and a HMM to integrate the residual temporal correlation. The GMM is calibrated over a learning set of continuous valued decision variable time series coming directly from observations. Clearly, using the multi-dimensional Kalman innovation process, we form a *one-dimensional* residual vector, which we put as input of the GMM to build a family of gaussians. These normal distributions will be later transformed into discrete sequences using a Maximum A Posteriori criterion. Thereafter the HMM is calibrated over the discrete sequence of GMM Components membership, for time dependence learning. The three new components consists of a MAP (Maximum A Posteriori) phase that maps each observation to an index of the discrete mixture using a Maximum a Posteriori probability; thereafter a Viterbi algorithm is used to detect and **select** the state of the decision variable. Finally in the third and last step, threshold based anomaly detection is only applied if we are in these states detected by the Viterbi path. We will describe in detail these three steps in the forthcoming.

III. Methodology

A. Normal behavior modeling

The first step is to seek for a normal behavior model that captures traffic dynamics. For this sake, two classes of models have been proposed in the literature: PCA-based model and state-space model. In PCA-based method, we build the predictive model assuming the projection in an orthogonal subspace obtained through application of PCA in a good predictor of the signal. We refer the reader to a precise description of this class of model in [1]. State-space model is a classical approach to model dynamical signal and is shown in Eq. 1.

$$\begin{cases} X_{t+1}=C_tX_t+W_t \\ Y_t\ =A_tX_t+V_t \end{cases} \tag{1}$$

The model contains two equations: the first describes the dynamic of state variations and the second one the measurement dynamics that may result, in not directly observing the states but rather a linear combination of them. The noise W_t accounts for intrinsic noise in the state variations as well as modeling errors; the noise V_t accounts for measurement errors. Both are assumed to be uncorrelated zero-mean gaussian white-noise processes with covariance matrices Q_t and R_t, respectively. This class of models can be used to model a very large class of signals and in particular, they can be calibrated to model any signal if the number of states is large enough [4].

Normal behavior model calibration can be done for PCA models using the diagonalisation of the observation covariance matrix [1]. The calibration of the state-space model is presented in [4].

B. Decision variable generation

After calibrating the normal behavior model over a learning set, we can use this model to generate the decision variable. The decision variable in PCA is derived as the *square sum of the prediction error*, where the prediction error is derived as the difference between the observation and the projection in the PCA-based subspace; the reader can refer to [1] for an in detail analysis of the decision variable generation for PCA.

For state-space based approach, the decision variable is obtained through the application of a Kalman filter. The decision variable is derived as the *sum of squares of the innovation process weighted by its variance*. When the state space is a n-dimensional, the resulting decision variable becoming a χ^2 random variable with n degree of freedom. When n is large enough, this converges to a gaussian random variable with mean n and variance $2n$.

C. Kalman filter equations

The first problem to solve after building a simple model to monitor features (link counts and TCP/UDP metrics), is to find an optimal estimate ($\hat{X}_t$) of our unobservable network states X_t, given a set of past and current observations $\{Y_1, ..., Y_t\}$. To estimate the state of the system using only all information until time t, a robust method is the Kalman fixed-interval filtering algorithm. From the dynamical linear system, we refer to Y_t as the observation vector at a specific time t. And the state of the system at time t is given by X_t; let also $\hat{X}_{t|k}$ denotes the estimate of X_t using all the information available up to time k, i.e, $\forall \tau < k$. $\hat{X}_{t+1}$ denotes the estimate of X_{t+1} using all the information up to time t, (this constitutes the *phase predictor*). The quantity $\hat{X}_{t+1|t+1}$ denotes the estimate of X_{t+1} using all past information and the recently arrived data point at time t+1. On the other hand, $P_{t|t}$ denotes the variance of the *state estimate* and $P_{t+1|t}$ indicates the variance of *the state prediction*. As it is shown in its earlier elaboration, the Kalman filter addresses the problem of estimating a discrete state vector when the observations are only a linear combination of the underlying state vector. As an iterative algorithm, it estimates the system state using two steps: the filter runs as a *predictor-corrector* algorithm. Prediction comes in the *time update* phase, and correction in the *measurement update* phase.

- **Prediction step** (*time update equations*):
  In this step, the estimated state of the system at time t, $\hat{X}_{t|t}$, is used to predict the state at next time t+1, $\hat{X}_{t+1|t}$. And, as we know that the noise W_t influences the evolution of the system at each time t, we compute only the variance of the prediction, $P_{t+1|t}$ based on the updated variance at the previous time t, $P_{t|t}$, and the noise covariance at the same time, Q_t. The error covariance $P_{t+1|t}$ provides an indication of the uncertainty associated with the state estimate.

$$\begin{cases} \hat{X}_{t+1|t}= \quad C_t\hat{X}_{t|t} \\ P_{t+1|t}=C_tP_{t|t}C_t^T+Q_t \end{cases} \tag{2}$$

- **Correction step** (*measurement update equations*):
  This step updates (corrects) the state and the variance of the estimate in the previous step, using a combination of their predicted values and the new observations Y_{t+1}. The accuracy of this update depends on the Kalman innovation $Y_{t+1} - A_{t+1}\hat{X}_{t+1|t}$.

$$\begin{cases} \hat{X}_{t+1|t+1} = \hat{X}_{t+1|t} + K_{t+1}(Y_{t+1} - A_{t+1}\hat{X}_{t+1|t}) \\ \quad P_{t+1} = (I - K_{t+1}A_{t+1})P_{t+1|t}(I - K_{t+1}A_{t+1})^T \\ \qquad\qquad + K_{t+1}R_{t+1}K_{t+1}^T \end{cases} \tag{3}$$

In the measurement equations, K_{t+1} denotes the Kalman gain. For more details in linear dynamical system, estimation and Kalman filtering techniques, we refer the reader to : [5],[6],[7] and [8]. The above equations with initial conditions of the state of the system $\hat{X}_{0|0} = E[X_0]$ and the associated error covariance matrix $P_{0|0} = E[(\hat{X}_0 - X_0)(\hat{X}_0 - X_0)^T]$ define the discrete-time sequential recursive algorithm for determining the linear *minimum variance* estimate known as the *Kalman filter*.

D. Gaussian Mixture Model

In Kalman filtering philosophy, it is more generally assumed that the residual remains a zero mean gaussian process, however this assertion is not always true in practice. Thus our motivation in using gaussian mixture model is based on our belief that the real distribution of the process is an ensemble (mixture) of gaussians and there is some time dependency in the innovation (which we will later study by using hidden markov model theory). This assertion allows us to build a method, which has the ability to find anomalies in different families built on a one-dimensional residual process. The GMM takes as input the 1-dimensional residual vector we formed with the multi-dimensional Kalman innovation process, and finds a few number (K) of families (clusters). Each gaussian component is fixed by its first order statistic. The variance obtained for each component will help to determine the suitable number of K. Each cluster contains data coming from one gaussian component. Thereafter, we aim to convert each cluster into discrete sequence of symbols (1,2,3,...) by means of a MAP criterion. In the next step, we propose the use of an hidden markov model (HMM) to classify these discrete groups into P states. Each hidden state could probably contain mixing symbols yielding in different clusters. This operation has the main advantage to discover the potential temporal dependencies in the innovation process.

Technically, to find the values of the model parameters μ_m and Σ_m , as well as the prior probability vector π, we are interested in maximizing the *likelihood* $\pounds(\theta|\mathbf{X}) = p(\mathbf{X};\theta)$ of generating the known observed data (X) given the model parameters $\theta = \{\mu_m, \Sigma_m, \pi_m\}$, 1≤m≤M. **X** denotes all the observation while θ contains all the parameters of the mixture. In other words, we hope to find $\hat{\theta}_{ML}$=argmax $p(x|\theta)$. This approach is called the *Maximum Likelihood* (ML) framework since it finds the parameter settings that maximize the likelihood of observing the data sets. To find the best parameters of the features of θ, the Expectation Maximization (EM) iterative algorithm can be used to simplify the mathematical routines considerably and numerically compute the unknown parameters. In the **E-step** (*Expectation phase*), the parameters are estimated given the observed data and current estimates of the model parameters (i.e EM comes with an initial guess of the model parameters, μ_m, Σ_m and π_m; we have used the K-means algorithm). In the **M-step** (*Maximization phase*), EM takes the expected complete log-likelihood and maximizes it w.r.t. the parameters to estimate (π_m, μ_m, Σ_m). For more details about mathematical routines for EM, see [9],[10],[11].

E. Hidden Markov Model

From the above learning phase, each GMM component is transformed into a sequence of a finite set of alphabet (symbols as 1,2,3,...), using a *maximum a posteriori* criterion. This discrimination phase will help for plugging the above clusters into different a priori unknown states, using hidden markov model. We represent these families of states by the following collection of unknown random variables $\{Q_1, Q_2, ...Q_T\}$ (where Q_t is a constant value with values in $\{1, 2, ..., K\}$). We also represent our alphabet by the known vector $\{O_1, O_2, ..., O_T\}$. Now, the problem is resumed to find a model to produce the states and to determine the probability of each symbol being in a state.

A well known model-based approach to tackle this problem, is the discrete *hidden markov model* (HMM) approach. Our choice of using HMM is based on the fact that: i) potential time dependencies in the innovation process can be modelled and captured using a finite set of a *priori hidden states*, each of them containing a subset of gaussian components ii) relatively efficient algorithm can be derived to solve the problems related to them, [9], [10], [12]. The full HMM model we used is defined by the quantity $\lambda = (A, B, \pi)$ (where A is the *transition matrix*, B is the *emission probabilities* matrix and pi the *prior* probabilities).

To find and estimate the best parameters of our model, we use the well-known *forward-backward* algorithm parameter estimation (or Baum-Welch algorithm). For more details for EM techniques related to hidden markov model, see, [9], [12]. Thereafter, we reuse the model to find the optimal state sequence associated with the given observation sequence. We believe that this final step of our approach will allow us to capture all the variations in the innovation process. An optimal criterion we have chosen here is to find the single best state sequence (path), $Q = \{q_1, q_2, ..., q_T\}$ for the given observation sequence $O = \{O_1, O_2, ..., O_T\}$,i.e., we aim to maximize $P(Q|O, \lambda)$. A formal technique for finding this unique best state sequence is the Viterbi algorithm. The Viterbi algorithm will find a unique path (containing some of the symbols) witch will be able to capture all the variations in the decision variable.

IV. Model Evaluation

A. Experimental Data

In this work we used two kinds of data coming from two different networks: the Abilene and the SWITCH networks. The Abilene backbone has 11 Points of Presence(PoP) and spans the continental US. The data from this network was collected from every PoP at the granularity of IP level flows. The Abilene backbone is composed of Juniper routers whose traffic sampling feature was enabled. Of all the packets entering a router, 1% are sampled at random. Sampled packets are aggregated at the 5-tuple IP-flow level and aggregated into intervals of 10 minute bins. The raw IP flow level data is converted into a PoP-to-PoP level matrix using the procedure described in [2]. Since the Abilene backbone has 11 PoPs, this yields a traffic matrix with 121 OD flows. Note that each traffic matrix element corresponds to a single OD flow, however, for each OD flow we have a seven week long time series depicting the evolution (in 10 minute bin increments) of that flow over the measurement period. All the OD flows have traversed 41 links. Synthetic anomalies are injected into the OD flows by the methods described in [2], and this resulted in 97 anomalies in the OD flows.

The second collection of data we used for our experiments is a set of three weeks of Netflow data coming from one of the peering links of a medium-sized ISP (SWITCH, AS559),[13], [1]. This data was recorded in August 2007 and comprise a variety of traffic anomalies happening in daily operation such as network scans, denial of service attacks, alpha flows, etc. For computing the detection metrics, we distinguish between incoming and outgoing traffic, as well as TCP and UDP flows. For each of these four categories, we computed seven used traffic features: *byte, packet, flow counts, source and destination IP address entropy, unique source and destination IP address counts*. All metrics were obtained by aggregating the traffic at 15 minute intervals resulting in 28x96 data matrix per measurement day. Anomalies in the data were identified using available manual labelling methods, as visual inspection and time series and top-n queries on the flow data. This resulted in 28 detected anomalous events in UDP and 73 detected in TCP traffic. The SWITCH data was collected in a single link and the observed metrics are correlated in time and in space.

B. Validation

1) Tuning of the System parameters.: Our method begins with a learning phase where we calibrate a Kalman filter for denoising, using our linear dynamical system (1), and the collection of observation data . In order to run the Kalman filter, we need the state and measurement matrices A, Q, C and R. For the Abilene data, the matrix A is available given the routing scheme of a network. We thus only need to obtain Q, C and R. It is sufficient to learn the model's parameters using a sample of one week measurements. We find the tuning parameters , using two days of consecutive samples of all link counts, and we do our calibration using the *EM algorithm* developed in [14],[15] and implemented (in R language) in [16]. In Practice to run and find the GMM and HMM parameters, we had used the HMM toolbox [17].

The way to find the tuning parameters for the Kalman filter for the SWITCH data, is slightly different from the case of the Abilene case, because for these data, we don't know a priori the matrix A and it should be estimated. For this case, we used the EM algorithm developed in [18], and implemented (in Matlab) in [19].

To build the model for the PCA and the KL expansion, we use the vector of metrics $X[1:192]$ and the vector of link counts $X[1:288]$ containing the first two days of data, respectively for the SWITCH network and for the Abilene backbone. And we follow exactly the method described in [1] (based on pole-zero diagram) to choose the components number to be included in the model.

To apply the KL expansion, the SVD (Singular Vector Decomposition) technique is applied to the data (resulting to a basis change matrix) and the squared prediction error $Q[k] = e[k]^T e[k]$ ($e[k]$ is the residual process)is computed from, which the decision variable D[k] is calculated. Thus, using the multi-dimensional data (all vectors of metrics and all vectors of link counts), we obtained the *one-dimensional* array $D[k]$ with, which we perform anomaly detection.

As for the KL expansion, the same samples of metrics and link counts are used for the calibration of the Kalman filter. Thereafter we use the multi-dimensional innovation process to form a *one-dimensional* innovation process used to perform anomaly detection. To obtain this one-dimensional process, we take into account the variance of the residual obtained after running the Kalman filter and we built a new process using the formulas: $e_{new}(t) = e(t)^T V e(t)$, where V is the inverse of the variance of the innovation process, e(t) is the multi-dimensional innovation process, and T denotes the transpose. This space reduction for the residual process allows to perform a good comparison between the PCA-KL expansion method and our method based on Kalman filtering. On the other hand, performing anomaly detection in a single one-dimentional residual vector is more simple and less complex than analysing a multi-dimensional array.

2) Summary of the results.: The first result to show is the ability of our method to track the behavior of link counts for the Abilene data (total byte per unit time) and the behavior of the different TCP and UDP metrics for the SWITCH data, over time. In Figure 2, we show the real and inferred link counts (Abilene) for our model. The evolution of the traffic and estimates are shown for a seven weeks duration for each observation vector. The calibration is applied only once. In Figure 3, we show the results obtained using SWITCH data for the TCP metrics; here too, the calibration is done once a time, for a three weeks of measurements. The results for the UDP metrics are quite similar.

Now we are looking at the compared performances of our two methods, for the Abilene network and also for the SWITCH network. First, to validate our model, one has to find the suitable number of components in the GMM. To do this, we calibrate a GMM with a set of r components (r=2,3,4,5...) using the EM algorithm as described above, and the decision to select the best model (i.e the suitable r) is done by analysing the variance performed for each component in the mixture. We compare the results obtained for the different values of r and the model with the lowest variance is chosen.

We have found that the data residual can be organized into three (r=3) distinct clusters for the Abilene data, and into five (r=5) clusters for the SWITCH data. Thereafter *a maximum a posteriori* criterion is used to build, taking as input the clusters, a finite alphabet of symbols, where we perform the *hidden markov modeller*. To train the *hidden markov model*, one must ensure that the different *hidden states* are clearly distinct. This means that one should have a *transition matrix* with higher probabilities in its **main diagonal**. For the datasets we used, we have discovered that an hidden markov model with two (2) states were able to capture the temporal dependencies in the datasets. We show below only the transition and observation matrices for the Abilene case:

$$transmat = \begin{bmatrix} 0.9662 & 0.0338 \\ 0.0162 & 0.9838 \end{bmatrix}$$

$$obsmat = \begin{bmatrix} 0.0113 & 0.0020 & 0.9867 \\ 0.4092 & 0.5904 & 0.0004 \end{bmatrix}.$$

These results show clearly that, the state 1 is composed with almost entirely the symbol$\#3$ and the state 2 is a mixture of the two remaining symbols (1 and 2) with 41$\%$ of probability of presence of symbol$\#1$ and 59$\%$ for symbol$\#2$. In the philosophy of anomaly detection theory, generally it is assumed, that anomalies might be rare; base on this assumption we can ask this question: if anomalous events occur, do they might come from state $\#1$ or state $\#2$ or both? It is not obvious to answer this question, but we believe that all the changes in the mean of the residual (abrupt changes, lower or higher variations) can be tracked by a combination of symbols yielding in the different states. In other words, one can think that some (but not all) a priori unknown states could be classified as *normal* states and the others as

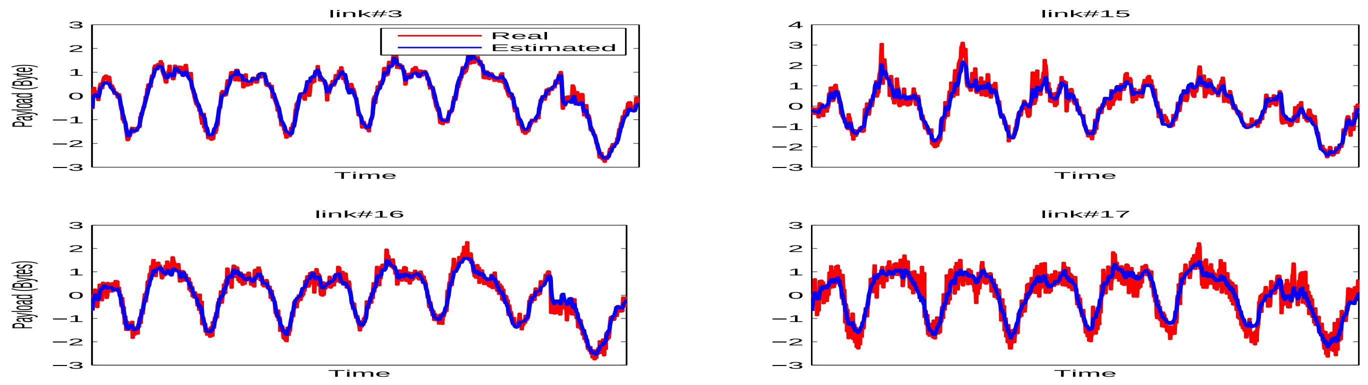

Fig. 2. Real(red) and estimated (blue) link counts obtained using Kalman filter.

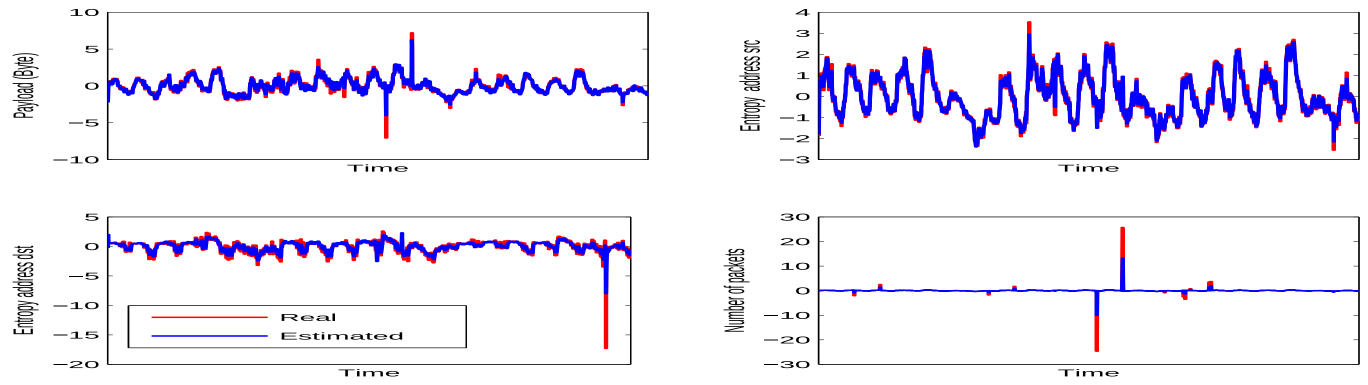

Fig. 3. Real(red) and estimated (blue) TCP metrics obtained using Kalman filter.

abnormal states. If one can find a combination of states to track all the lower, higher or/and abrupt variations in the decision variable, these states will participate in the detection of anomalous events when they occur. And then these states will be etiquetted as abnormal, and one should take attention to them. The remaining states that don't participate in the tracking operation will then be labelled as normal. Recall that PCA and Kalman-based anomaly detection techniques analyze residuals to perform the detection issue. So, if anomalies occur, they will appear in the residual process either in the form of low or high variations or in the form of abrupt changes in the mean of the residual process. We believe that it will be interesting to divide the residual into two parts corresponding to the low and high variations, and ***separately*** apply thresholds for each part. To confirm our intuition, we run the Viterbi algorithm for all the sequences of discrete alphabet and we obtain one unique sequence (path) composed by only the symbols in the state #2.

At this time, we can argue that, if anomalies exist they might be caught either by the two symbols simultaneously, or by one symbol only. One can observe that in Fig. 4, all the variations in the mean of the decision variable can be caught by these two symbols. The top graph corresponds to TCP, the middle to UDP and the bottom graph to Abilene. To track the anomalies, one has just to extract the residual corresponding to the symbol #1 and extract also the part of residual corresponding to symbol #2 and apply to each part thresholds. We reused the methods described in [1] to obtain these thresholds. In addition, in our study we discover that the injected anomalous events never evolve in the cluster with mean closely equal to zero(namely the cluster corresponding to symbol #3).

Another result is about the performance obtained in analyzing the trade-off between false positive and false negative rates. We examine the entire traffic for each method, namely the PCA-KL and the Kalman-based approaches. We then can compute one false positive percentage and one false negative percentage for each threshold configuration scheme. The performances of the two methods on the Abilene and the SWITCH data are depicted in the ROC (Receiver Operating Characteristic) curve of Figures 5, 6 and 7. For all of the results shown on these figures, one can see clearly that the method based on the Kalman fitering techniques, the gaussian mixture model and the hidden markov model performs better than the KL-PCA method, when the temporal correlation range is set to N=1,2,3. For the TCP traffic (SWITCH network), we obtain in Figure 5, 90% of detection rate with 8% of probability of false alarm, for KL expansion with N=2, while we have only 3.5% of false alarm rate with the same detection probability for our new method with N=2, and for N=3, we obtain 2.5% of false positive rate. We observe that, with the newer approach, the false alarm ratio decreases significantly

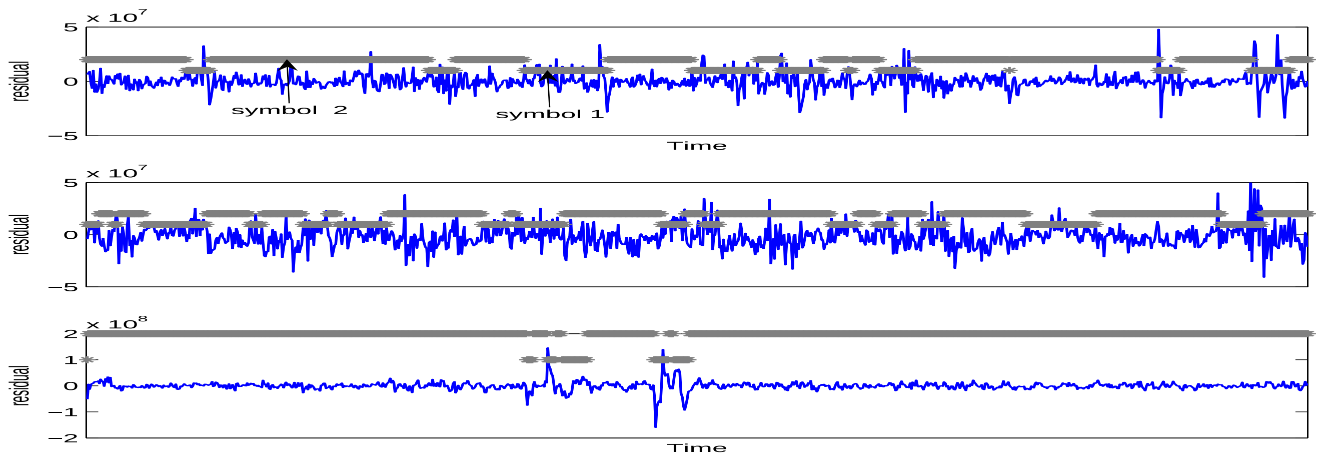

Fig. 4. Tracking the low and high variations in the decision variable by combination of mixing symbols extracted by the Viterbi algorithm.

for all values of N. In the same figure we note that the newer method can achieve 100% (0% of miss detection) of probability of detection with 3.2% of false positive rate while the KL expansion exhibits the same detection rate with 14% of false alarm rate. Also, for the UDP traffic, we obtain in Figure 6, for the KL expansion method for N=2, 96% of detection rate with 7% of false alarm rate versus 2.4% (for N=3) of false alarm rate with the same probability of detection. As for the TCP case, here too, our new method can achieve 100% of detection rate with 2.5% of false positive (N=3) when the KL expansion shows 12% of false positive with the same detection rate (for N=2).

For the Abilene network, we confirm the improvement in the performance of our method above the PCA-KL expansion. One must clearly observe, in Figure 7, that the KL expansion (N=2) shows 100% of probability of detection with 13% of false positive rate while the new method exhibits a false alarm rate of 5% (N=3) with the same detection rate. In summary, the ROCs curve exhibit different points, which can be served as good references to show the high performance we gain above the KL method.

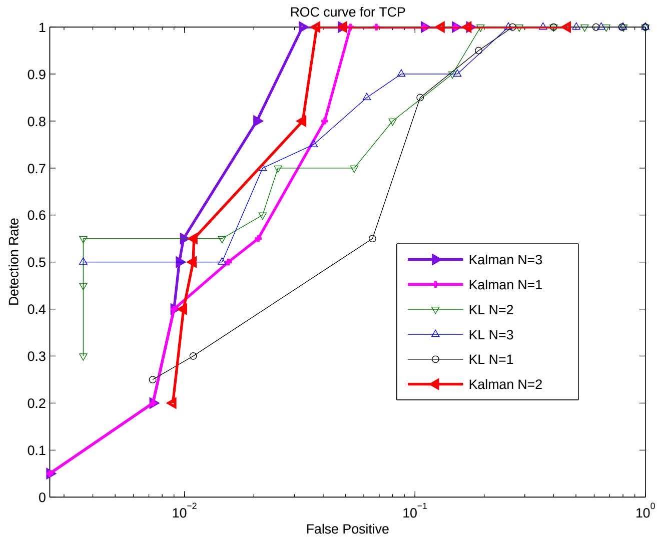

Fig. 5. ROC curve using SWITCH data (TCP)

V. Conclusion

In this work we proposed a profound analysis allowing us to show that an anomaly detection technique combining a panoply of different methodologies and based on Kalman filtering perform better than the PCA technique, which the performance is highly improve by the use of the Karuhen-Loeve expansion. Using a multi-dimensional residual process for each kind of network data, we built a one-dimensional innovation process used as a decision variable, and the comparison of the two schemes is done by analyzing the trade-off between false positive rate and the probability of detection. We found that the use of the Viterbi algorithm as a final tool to make possible to split the one-dimensional decision variable into several subset where we applied different thresholds is an important discovery. It has make possible to track the variability in the residual process using only a combination of symbols yielding in one state. The main drawback of

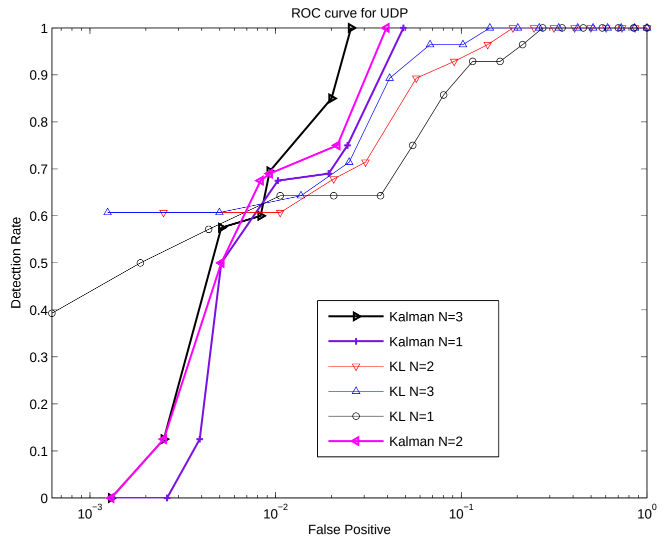

Fig. 6. ROC curve using SWITCH data (UDP).

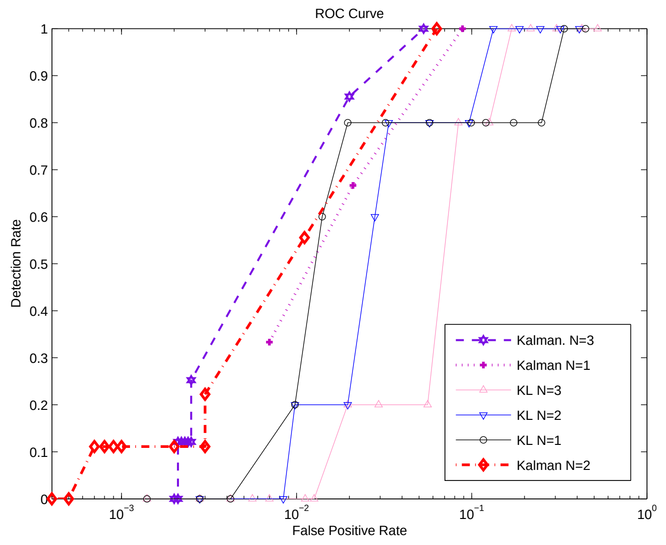

Fig. 7. ROC curve using Abilene data.

this method is the relative complexity introduced by the use of a gaussian mixture model followed by using an HMM, for time dependencies tracking. However, we obtained good performance by reducing considerably the false alarm rate. We had shown in this study that for the two kind of data, the temporal dependencies can be tracked with a hidden markov model with a few number of states (each state being composed by parts of the GMM component). At the other hand, when applying the Viterbi algorithm, we discover that all anomalies are detected in one state with only components 1 and 2, no anomaly were found in the remaining components (number 3 for Abilene and number 3 ,4 and 5 for SWITCH network). The state where no anomaly has been found is the one containing the clusters with mean closely equal to zero. At this moment, one could ask a question about the potential uncertainties about the possibility of these remaining components to capture anomalies in some situations.

REFERENCES

[1] Brauckhoff, D., Salamatian, K. and May, M.: Applying PCA for Traffic Anomaly Detection: Problems and Solutions. Proceedings IEEE INFOCOM 2009, pp. 2866-2870.
[2] Lakhina, A., Crovella, M. and Diot, C.: Characterization of network-wide traffic anomalies. In Proceedings of the ACM/SIGCOMM Internet Measurement Conference (2004). pp. 201-206.
[3] Soule, A., Salamatian, K. and Taft, N.: Traffic Matrix Tracking using Kalman Filters. ACM LSNI Workshop (2005).
[4] Soule, A., Salamatian, K. and Taft, N.: Combining Filtering and Statistical Methods for Anomaly Detection. USENIX , Association, Internet Measurement Conference (2005). pp. 331344.
[5] Kalman, R. E. and Bucy R. S.: New results in linear filtering and predictions. Trans. ASM E., Series D, Journal of Basic Engineering, Vol. 83 (1961), pp. 95-107.
[6] Kailath, T., Sayed, A. H. and Hassibi B.: Linear Estimation. Prentice Hall, 2000.
[7] Wolverton, C.: On the Linear Smoothing Problem. IEEE Transactions on Automatic Control, vol. 14 Issue:1 pp. 116-117. February 1969.
[8] Raugh, H. E.: Solutions to the linear smoothing problem. IEEE Trans. Automatic Control AC-8 (October 1963) pp. 371372.
[9] Bilmes, J. A.:A Gentle Tutorial of the EM algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models, International Computer Science Institute, Berkeley CA. Technical Report TR-97-021, ICSI, 1997.
[10] McLachlan, G. and Krishnan, T.: The EM Algorithm and Extensions. John Wiley and Sons, New York, 1996.
[11] Dempster, A. P., Laird N. M., and Rubin D. B.: Maximum likelihood from in-complete data via the em algorithm. Journal of the Royal Statistical Society: Series B, 39(1): pp. 138, November 1977.
[12] Rabiner, L., R.,: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proc. IEEE, Vol. 77, No. 2, pp. 257-286, February 1989
[13] Brauckhoff, D., Dimitropoulos, X., Wagner, A. and Salamatian, K. : Anomaly Extraction in Backbone Networks using Association Rules. IMC09, November 46, 2009, pp. 28-34 Chicago, Illinois, USA.
[14] Shumway, R. H. and Stoffer, D. S.: An Approach to Time Series Smoothing And Forecasting Using the EM Algorithm. Journal of Time Series Analysis, vol.3, No 4,pp. 253-264.
[15] Shumway, R. H. and Stoffer, D. S.: Dynamic Linear Models With Switching. Journal of the American Statistical Association, 86, pp. 763-769, 1992.
[16] http://www.stat.pitt.edu/stoffer/tsa2/chap6.htm. 2011
[17] http://www.cs.ubc.ca/ murphyk/Software/HMM/hmm.html. 2011
[18] Ghahramani, Z. and Hinton, G. E.:Parameter Estimation for Linear Dynamical Systems. Technical Report CRG-TR-96-2. February 22, 1996.
[19] http://learning.eng.cam.ac.uk/zoubin/software.html. 2011