International Journal of Control and Automation
Vol.7, No.3 (2014), pp.389-400
http://dx.doi.org/10.14257/ijca.2014.7.3.37

Automated Surveillance in Distributed, Visual Networks: An Empirical Comparison of Recent Algorithms

Tarem Ahmed
Department of Electrical & Electronic Engineering
BRAC University
Dhaka, Bangladesh
tarem@bracu.ac.bd

Supriyo Ahmed
Department of Electrical & Electronic Engineering
BRAC University
Dhaka, Bangladesh
supriyo@bracu.ac.bd

Al-Sakib Khan Pathan
Department of Computer Science
International Islamic University Malaysia
Kuala Lumpur, Malaysia
sakib@iium.edu.my

Abstract

A number of algorithms have been recently proposed for automatic intruder detection from CCTV images. Past researchers have typically tested these algorithms on centralized networks where all images are transmitted to a central control room. This paper demonstrates the applicability of a selection of such algorithms to a distributed network of wireless sensors. A distributed network of wireless visual sensors was simulated using a number of high-resolution webcams setup in the hallways of an academic building. The selected algorithms were then applied in a distributed fashion at each node. An empirical comparison of the most popular of the recent algorithms on a simulation of a wireless sensor network was thus obtained. This paper provides corroborating evidence in support of the most effective of such algorithms to the problem of automatic anomaly detection from image streams.

1 Introduction

Physical security is unfortunately of prime concern in today's world, and an extensive network of multimodal surveillance networks is prevalent in many places. They range from analogue closed-circuit television (CCTV) systems to sophisticated networks of infra-red and motion sensors in sensitive areas such as banks and museums.

Ahmed et al. have recently proposed three schemes based on kernel machines to perform automated detection of unnatural activity in visual surveillance systems [1]. Ahmed et al. have compared their proposed algorithms with two representative schemes selected from two families of methods popularly used in automated surveillance. They have tested on an infrastructure consisting of a centralized network of archaic CCTV cameras around a poorly-lit building.

In this paper, we argue the applicability of the algorithms introduced in [1] to a distributed network of wireless visual sensors. Minimizing communication costs is imperative in a mobile and wireless network, unlike in a static, centralized network that can use backbone wired connections such as a LAN. We simulate a network of wireless visual sensors by using a network of high-resolution webcams deployed randomly inside a building, and apply the algorithms discussed in [1] in a distributed manner at each node.

This paper also provides a comparative empirical comparison of the algorithms discussed in [1] on a *complementary* setting, thus providing corroborating evidence in support of the kernel-based algorithms first advocated in [1] to the general problem of automated intruder detection in surveillance networks.

1.1 Organization of Paper

The rest of this paper is organized as follows. Section 2 outlines the algorithms that have been selected from the recent literature. Section 3 describes the experimental setup and discusses the experimental results. Section 4 concludes with suggestions for future work.

2 Algorithmic Bases

2.1 Kernel Functions

Algorithms based on the so-called "kernel trick" involve using a *kernel* function that maps the input data onto a *feature space* of much higher dimension, with the expectation that points depicting similar behaviour would cluster in the richer, higher-dimensional (feature) space [2]. A suitable kernel function, when applied to a pair of input vectors, may be interpreted as an inner product in the feature space. This subsequently allows inner products in the feature space (inner products of the *feature vectors*) to be computed without explicit knowledge of the feature vectors themselves, by simply evaluating the kernel function:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \tag{1}$$

where $\mathbf{x}_i, \mathbf{x}_j$ denote the input vectors and ϕ represents the mapping onto the feature space.

2.2 Kernel-based Online Anomaly Detection Algorithm

If the points $\{\mathbf{x}_t\}_{t=1}^{T}$ show normal behaviour in the input space, then the corresponding feature vectors $\{\phi(\mathbf{x}_t)\}_{t=1}^{T}$ are expected to (also) cluster [3]. Then, it should be possible to explain the region of normality in the feature space using a relatively small *dictionary* of *approximately* linearly independent elements $\{\phi(\tilde{\mathbf{x}}_j)\}_{j=1}^{m}$. Feature vector $\phi(\mathbf{x}_t)$ is said to be *approximately* linearly dependent on $\{\phi(\tilde{\mathbf{x}}_j)\}_{j=1}^{m}$ with approximation threshold ν, if the projection error δ_t satisfies:

$$\delta_t = \min_{\mathbf{a}} \left\| \sum_{j=1}^{m} a_j \phi(\tilde{\mathbf{x}}_j) - \phi(\mathbf{x}_t) \right\|^2 < \nu. \tag{2}$$

where $\mathbf{a} = \{a_j\}_{j=1}^{m}$ is the optimal coefficient vector [4].

Observe that (2) involves an L2 norm, which may be simplified exclusively in terms of the inner products of $\phi(\tilde{\mathbf{x}}_j)$ and $\phi(\mathbf{x}_t)$, and thus evaluated using the kernel function without explicit knowledge of the feature vectors themselves:

$$\delta_t = \min_{\mathbf{a}_t} \left\{ \mathbf{a}_t^{\mathrm{T}} \tilde{\mathbf{K}}_{t-1} \mathbf{a}_t - 2\mathbf{a}_t \tilde{\mathbf{k}}_{t-1}(\mathbf{x}_t) + k(\mathbf{x}_t, \mathbf{x}_t) \right\} \tag{3}$$

where $[\tilde{\mathbf{K}}_{t-1}]_{i,j} = k(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j)$ and $[\tilde{\mathbf{k}}_{t-1}(\mathbf{x}_t)]_j = k(\tilde{\mathbf{x}}_j, \mathbf{x}_t)$ for $i, j = 1 \ldots m_{t-1}$. The optimum *sparsification* coefficient vector $\mathbf{a}_t$ that minimizes δ_t at time t is then:

$$\mathbf{a}_t = \tilde{\mathbf{K}}_{t-1}^{-1} \cdot \tilde{\mathbf{k}}_{t-1}(\mathbf{x}_t). \tag{4}$$

The expression for error δ_t may then be simplified into:

$$\delta_t = k_{tt} - \tilde{\mathbf{k}}_{t-1}(\mathbf{x}_t)^{\mathrm{T}} \cdot \mathbf{a}_t. \tag{5}$$

The Kernel-based Online Anomaly Detection (KOAD) algorithm operates at each timestep t on a measurement vector $\mathbf{x}_t$. It begins by evaluating the error δ_t in projecting the arriving $\mathbf{x}_t$ onto the current dictionary. This error measure δ_t is then compared with two thresholds ν_1 and ν_2, where $\nu_1 < \nu_2$. If $\delta_t < \nu_1$, KOAD infers that $\mathbf{x}_t$ is sufficiently linearly dependent on the dictionary, and represents normal behaviour. If $\delta_t > \nu_2$, it concludes that $\mathbf{x}_t$ is far away from the realm of normality and immediately raise a "Red1" alarm to immediately signal an anomaly. If $\nu_1 < \delta_t < \nu_2$, KOAD infers that $\mathbf{x}_t$ is sufficiently linearly independent from the dictionary to be considered an unusual event. It may indeed be an anomaly, or it may represent an expansion or migration of the space of normality itself. In this case, KOAD does the following: it raises an "Orange" alarm, keeps track of the contribution of the relevant input vector $\mathbf{x}_t$ in explaining subsequent arrivals for ℓ timesteps, and then takes a firm decision on it. Details regarding the KOAD algorithm are available in [3].

2.3 Kernel Density Estimation

Kernel Density Estimation is a popular nonparametric method of estimating the probability density function (pdf) of a random variable from a finite data sample. Given an independent, identically distributed sample set $\{\mathbf{x}_i\}_{i=1}^{n}$ drawn from an unknown probability distribution f, the value of pdf f at any point t may be estimated as:

$$\hat{f}(t) = \frac{1}{n} \sum_{i=1}^{n} k(\mathbf{x}_i, \mathbf{x}_t) \tag{6}$$

where $k(,)$ denotes the kernel function.

2.4 Kernel Estimation-based Anomaly Detection

The Kernel Estimation-based Anomaly Detection (KEAD) algorithm formally states the problem as follows [5]. Given a sequence of multidimensional data points $\{\mathbf{x}_i\}_{i=t-L}^{t+L} \in \mathbb{R}^D$, the objective is to determine if $\mathbf{x}_t$ is a realisation of probability distribution $P_{n,t}$ or of probability distribution P_a. It is assumed that the points $\{\mathbf{x}_i\}_{i=t-L}^{t+L} \in \mathbb{R}^D$ are independent observations from the mixture distribution P_t:

$$P_t = (1 - \pi)P_{n,t} + \pi P_a \tag{7}$$

where π is the mixing faction. The component distributions $P_{n,t}$ and P_a correspond to *normal* and *anomalous* traffic at time t, respectively. Distribution $P_{n,t}$ is assumed to be slowly time-varying, while P_a is time-invariant.

Assuming that the distribution governing the normal points, $P_{n,t}$, is stationary in $\{t-L : t+L\}$, leads to the following expression for the Kernel Density Estimate (KDE) at $\mathbf{x}_t$:

$$\tau_t = \frac{1}{2L} \sum_{i=t-L}^{t+L} k(\mathbf{x}_i, \mathbf{x}_t) \tag{8}$$

where $k(,)$ represents the kernel function. Now KDE τ_t is expected to be relatively low if $\mathbf{x}_t$ arises from the anomaly distribution P_a, compared to if $\mathbf{x}_t$ arises from the normal distribution $P_{n,t}$. KDE τ_t may thus be selected as a statistic for making a block-based (offline) anomaly decision on $\mathbf{x}_t$.

One may then use the dictionary $\mathcal{D}_{t-1}$ and the matrix $\mathbf{A}_t$ of optimal sparsification coefficient vectors $\mathbf{a_t}$ for the past L timesteps to obtain the *online* detection statistic $\hat{\tau}_t$:

$$\hat{\tau}_t = \frac{1}{L} \sum_{\ell=1}^{L} \sum_{j=1}^{m_{t-1}} \mathbf{a}_{\ell j} \cdot k(\tilde{\mathbf{x}}_j, \mathbf{x}_t) = \frac{1}{L} \sum_{i=1}^{L} \mathbf{A}_{t-1} \cdot \tilde{\mathbf{k}}_{t-1}(\mathbf{x}_t). \tag{9}$$

KEAD proceeds at every timestep t by first computing the optimum sparsification coefficient vector $\mathbf{a}_t$ from (4) and the projection error δ_t from (5), and the online detection statistic $\hat{\tau}_t$ from (9). The sparsification statistic δ_t is then compared with the sparsification parameter ν. If $\delta_t \geq \nu$, $\mathbf{x}_t$ is inferred to be approximately linearly *independent* of the space spanned by the dictionary at time t. Input vector $\mathbf{x}_t$ is consequently added to the dictionary. Contrarily if $\delta_t < \nu$, $\mathbf{x}_t$ is inferred to be approximately linearly *dependent* on the dictionary. The dictionary is then kept unchanged.

To make a decision regarding whether $\mathbf{x}_t$ is normal or anomalous, KEAD compares online detection statistic $\hat{\tau}_t$ with detection threshold η_t. If $\hat{\tau}_t \geq \eta_t$, the KDE of P_t at $\mathbf{x}_t$, that was computed using the dictionary and window of *past* L sparsification coefficient vectors, is high enough and $\mathbf{x}_t$ is inferred to represent normal traffic. Contrarily if $\hat{\tau}_t < \eta_t$, the KDE of P_t is low. In this case $\mathbf{x}_t$ either represents an anomaly, or $\{x_i\}_{i=t-L}^{t}$ is not a sufficiently representative sample of $\{x_i\}_{i=t-L}^{t+L}$ for the online detection statistic $\hat{\tau}_t$ to be an accurate estimate of true τ_t. The following is done in such a situation: an "Orange" alarm is raised at time t, $\mathbf{x}_t$ is stored for the next $\ell \ll L$ timesteps, and taking a firm decision on it is delayed for a further ℓ timesteps. Details regarding the KEAD algorithm are available in [5].

2.5 Principal Component Analysis

The technique of Principal Component Analysis (PCA) may be used to separate the space occupied by set of input vectors into two disjoint subspaces, corresponding to normal and anomalous behaviour [6, 7]. An anomaly may then be flagged in the timesteps where the magnitude of the projection onto the anomalous subspace, θ_t, exceeds a threshold. As

PCA is an iterative, block-based method, it may be applied to a real-time scenario using a sliding window method [1].

We include PCA in this comparative study for three primary reasons. First, employing PCA in this manner provides the simplest and most direct method of anomaly detection in a timeseries of multivariate measurements. Second, while kernel machines cluster the points in the *feature space* mapped onto by the chosen kernel function, PCA clusters in the *input space* itself, thereby providing a complementary approach. Third, PCA has been the basis for many recent autonomous intruder detection systems [1, 8].

2.6 Kernel Principal Component Analysis

Kernel Principal Component Analysis (KPCA) is an extension of standard Principal Component Analysis (PCA) that allows nonlinear regression [9]. Instead of determining the principal components of the input data vectors themselves as in standard PCA, Kernel PCA determines the principal components in the feature space mapped onto by the chosen kernel function. The standard PCA algorithm may be formulated in a form that involves only inner products, and thereby allowing substitution with kernel functions. The principal components are then found by solving the eigenvalue problem for the input vector *kernel matrix*, known as the Gram matrix. As KPCA is an iterative, block-based method, it may be applied to a real-time scenario using a sliding window method [1].

2.7 Normalized Compression Distance-based Similarity Metric

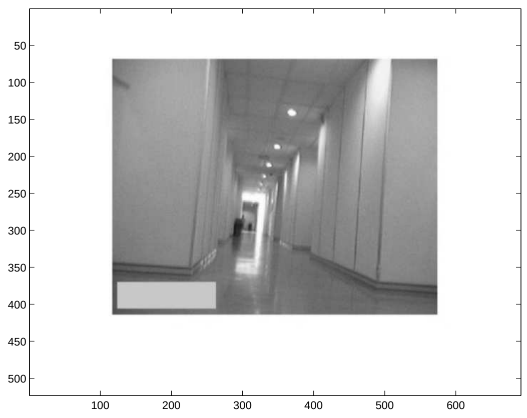
Au et al. have presented a celebrated algorithm in [10] where a set of novel images are stored, and arriving images are compared to this set. A scene is considered anomalous when the maximum similarity between the given scene and all previously viewed scenes is below a given threshold. Similarity is measured using the Normalized Compression Distance (NCD) measure. Au's algorithm first compresses each image individually, and then compresses concatenated pairs of images that are being directly compared. The similarity metric, ρ, is then evaluated comparing the compressed file size of the concatenated pair of images, and the compressed file sizes of the two images individually. The premise is that the size of the compressed version of the concatenation of two similar files should be *smaller* than that of the concatenation of two dissimilar files.

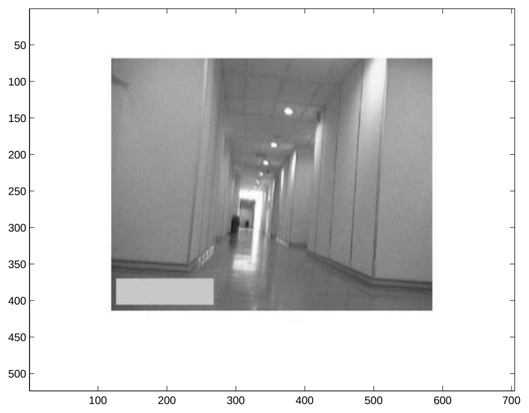
3 Experiments

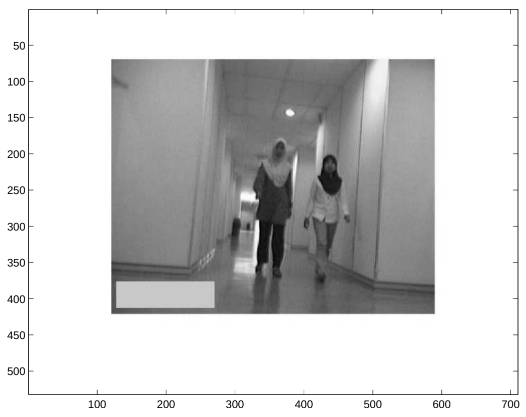
3.1 Data

In order to simulate a distributed network of visual sensors, four high-resolution Logitech$^{\mathrm{TM}}$ webcams were randomly deployed in a junction of hallways at the International Islamic University Malaysia (IIUM), and laptop computers connected to it were programmed to take still snaps every 15-seconds. The detection algorithms were then run individually at each node in a distributed fashion. The raw data thus comprised of still images in the JPEG format at 15-second intervals. The total data set consists of 300 timesteps, of which 27 were identified as potential anomalies after performing a manual inspection of the data set.

Figure 1 shows pictures from Camera 1 corresponding to four example timesteps. Within Fig. 1, subfigures (a) and (b) show regular (normal) scenarios, while subfigures (c) and (d)


(a) Normal image at $t = 100$.


(b) Normal image at $t = 200$.


(c) Anomalous image at $t = 93$.

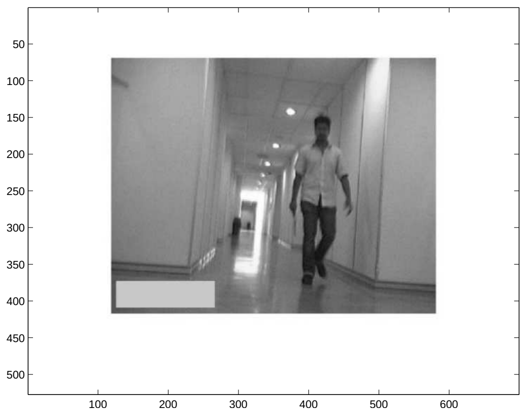
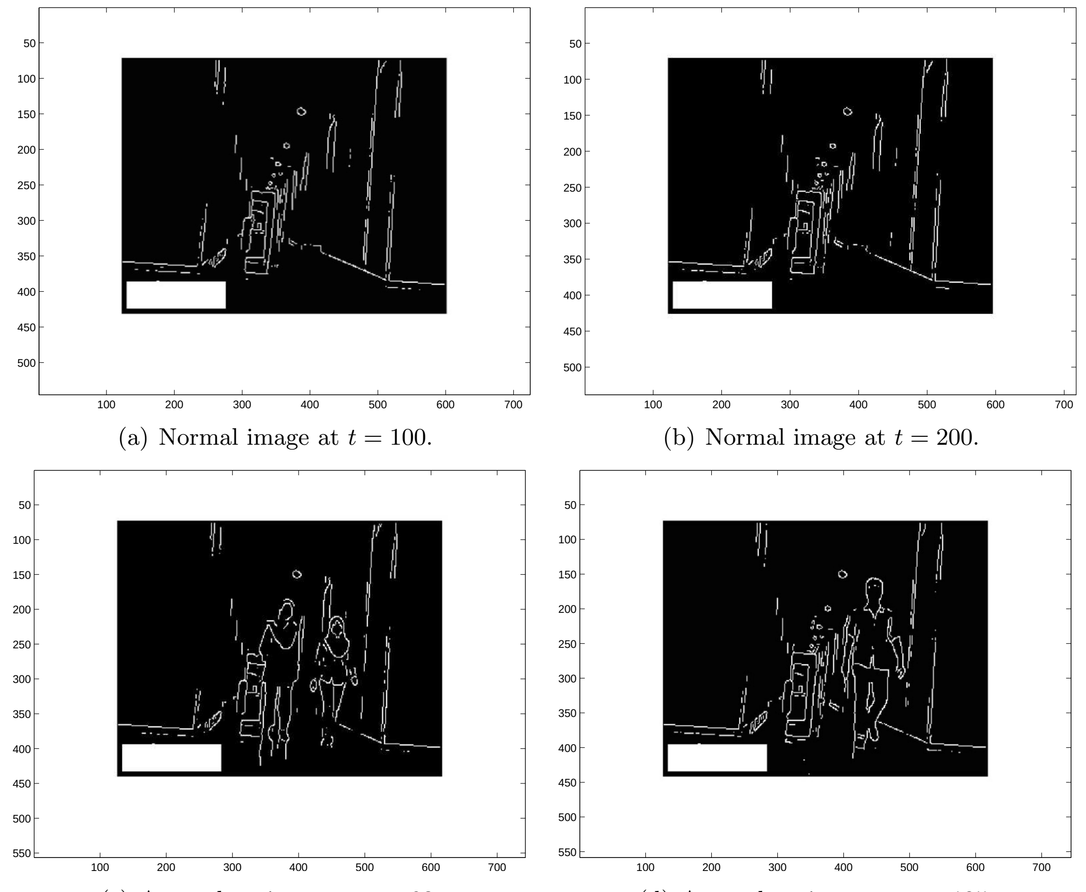

(d) Anomalous image at $t = 125$.

Figure 1. Images captured during four example timesteps at Node 1. Timesteps corresponding to (a) and (b) show usual images, while (c) and (d) show situations where humans appear. Actual time stamps on the images have been concealed for the sake of privacy.

show instances of human forms appearing on the scene. The actual timestamps have again been removed for the sake of privacy.

A high-resolution visual sensor captures a lot of details which may qualify as noise given our objective of detecting physical intruders. An elegant way of eliminating such noise is by using the Canny edge detector [10] to obtain an edge image where the step edges are enhanced. Figure 2 shows the Canny edge images corresponding to the example raw images from Fig. 1. The edge images instantly draw attention to the essential difference between subfigures (c) and (d) from subfigures (a) and (b), in the additional physical shapes present in subfigures (c) and (d).

After obtaining the Canny edge images (also in the JPEG format) at 15-second intervals, standard two-dimensional Haar wavelet decompositions [7, 11] were performed to obtain a 120×160 vector representation for each image at each camera, corresponding to each timestep. Transformation into the frequency domain is still necessary to account for differences between specific pixels in different images arising as a result of minor camera movements, and to consider and compare each image as a whole. Finally, 25% bilinear interpolation was performed to rescale and reduce the size of each dimension. This yielded



(a) Normal image at $t = 100$.

(b) Normal image at $t = 200$.

(c) Anomalous image at $t = 93$.

(d) Anomalous image at $t = 125$.

Figure 2. Canny edge images corresponding to the raw images from Fig. 1. Extra edges are visible in (c) and (d) compared to (a) and (b), corresponding to the extra physical forms present in (c) and (d).

one $30 \times 40 = 1200$-dimensional row vector of input data at each camera corresponding to each timestep.

3.2 Results

Figure 3 compares the performances of KOAD, KEAD and KPCA with PCA and Aus NCD-based algorithms. Receiver Operating Characteristic (ROC) curves are presented, demonstrating the tradeoff between the Probability of False Alarms (P_{FA}) and the Probability of Detection (P_D).

Default settings were used for all algorithms parameters [1]. It is evident from Fig. 3 that the performances of all of KPCA, KOAD and KEAD are significantly superior to those of PCA and NCD-based algorithms. Moreover, all kernel-based algorithms easily achieve near-perfect detection rates at low false alarm rates. KPCA perform relatively the best, as this is intrinsically a block-based algorithm, implemented here by using a sliding window, while KOAD and KEAD are intrinsically incremental algorithms. Next come KOAD and KEAD, the performances of both of which are observed to be comparable, with KOAD faring marginally better. It must be remembered here that the thresholds ν_1 and ν_2 must

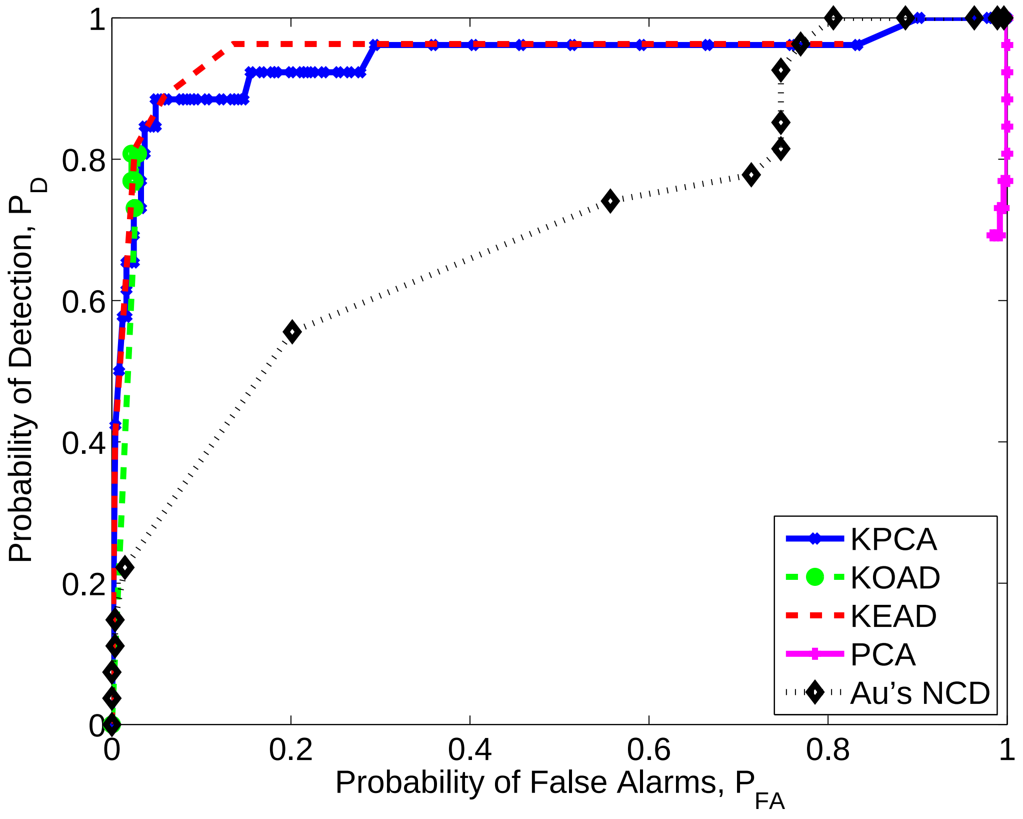
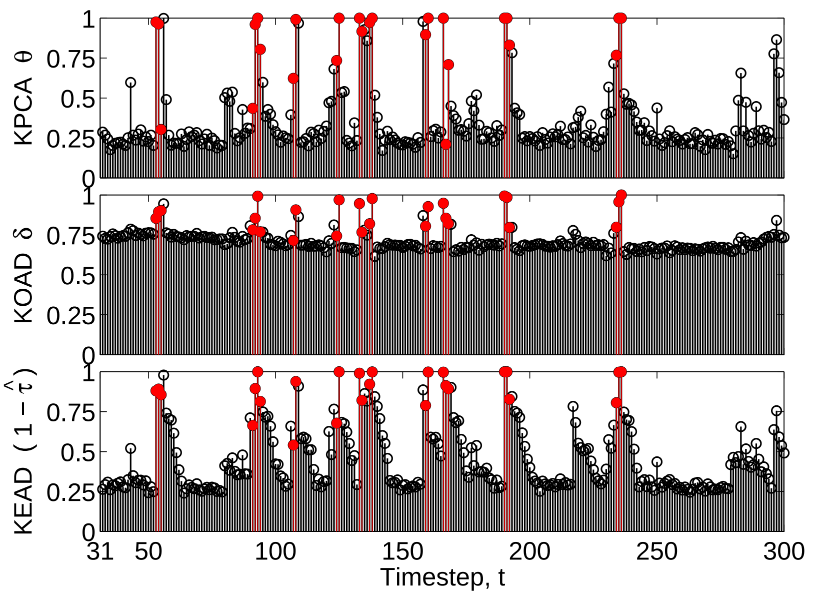


Figure 3. ROC curves showing performances of KPCA, KOAD, KEAD, PCA and Au's NCD-based algorithms. KPCA, KOAD and KEAD are observed to substantially outperform PCA and NCD

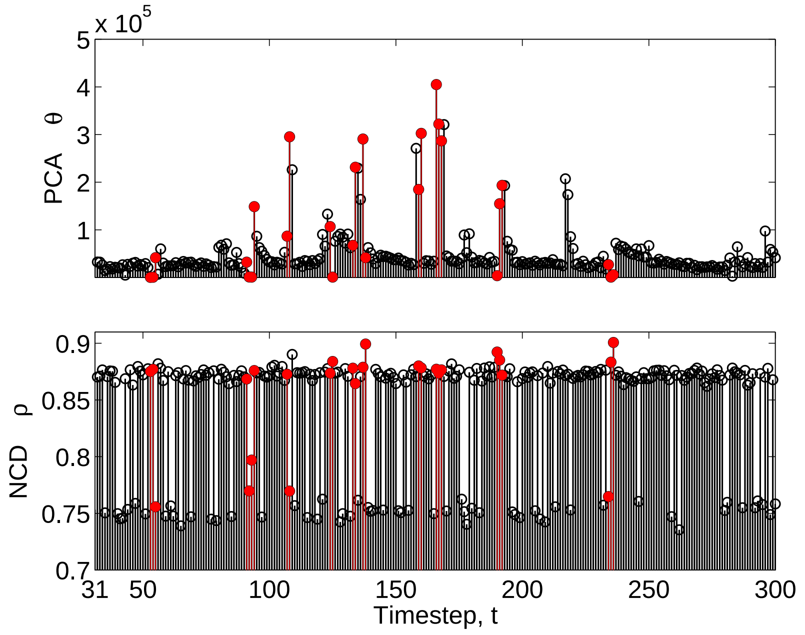
be manually set in KOAD, while KEAD incorporates autonomous setting for all important algorithm parameters [5]. The strikingly low performance of the benchmark NCD-based algorithm may be attributed to the fact that this algorithm requires a significantly longer training period, and needs to maintain a significantly larger database of images to compare new arrivals against [12].

Figures 4a and 4b present the detection statistics for each algorithm as a function of time. The location of the "true" anomalies, as were manually labelled, are indicated as red-filled circles.

It is clear from Fig. 4(a) that KPCA (top panel), KOAD (middle panel) and KEAD (bottom panel) do a good job of isolating the identified anomalies from the normal points. The example run of KOAD presented here was obtained using $\nu_1 = 0.10$ and $\nu_2 = 0.60$. It can be seen from Fig. 4(b) that PCA (top panel) misses a lot of the identified anomalies, while the performance of Aus NCD-based similarity metric (bottom panel) is clearly the worst. The inferences from Fig. 4 are thus in agreement with the ROC curves for the algorithms presented in Fig. 3.



(a) Progression in the anomaly detection statistics for KPCA, KOAD and KEAD algorithms for an example setting of the relevant detection thresholds. The true anomalies are indicated as red-filled circles. Top panel: Magnitude of projection onto the residual subspace, θ_t, for KPCA. Middle panel: KOAD projection error, δ_t. Bottom panel: $1-\hat{\tau}_t$ where $\hat{\tau}_t$ is the KEAD online detection statistic.



(b) Progression in the anomaly detection statistics for PCA and Au's NCD-based algorithms for an example setting of the relevant detection thresholds. The true anomalies are indicated as red-filled circles. Top panel: Magnitude of projection onto the residual subspace, θ_t, for PCA. Bottom panel: $1 - \rho_t$ where ρ_t is Aus NCD-based similarity metric.

Figure 4. Progression in the anomaly detection statistics for each algorithm for an example setting of the relevant detection thresholds.

4 Conclusion and Future Directions

Ahmed et al. have recently proposed three algorithms based on kernel machines to perform automated detection of unnatural activity in visual surveillance systems [1]. They have compared their proposed algorithms with two representative schemes selected from two families of methods popularly used in automated surveillance. They have tested on an infrastructure consisting of a centralized network of archaic CCTV cameras installed around a poorly-lit building.

This paper has demonstrated the applicability of the algorithms introduced in [1] to a distributed network of wireless visual sensors. In addition, this paper has also provided a comparative empirical study of the algorithms discussed in [1] on a complementary setting, thus providing further evidence in support of the kernel-based algorithms first advocated in [1] to the general problem of automated intruder detection in surveillance networks.

Our future work will concentrate on applying other machine learning algorithms such as the One-Class Neighbor Machine (OCNM) [13, 14] to the problem at hand, and applying the algorithms discussed here to intrusion detection problems other networks. Indeed, promising preliminary results have been obtained in applying the algorithms discussed here to malicious peer detection in peer-to-peer networks [15, 16], and in automatic congestion detection in road traffic surveillance systems [11].

References

[1] T. Ahmed, X. Wei, S. Ahmed, and A.-S. K. Pathan, "Efficient and effective automated surveillance agents using kernel tricks," *SIMULATION: Transactions of The Society for Modeling and Simulation International*, vol. 89, no. 5, pp. 562–577, May 2013.

[2] B. Schölkopf and A. Smola, *Learning with Kernels.* Cambridge, MA, USA: MIT Press, Dec. 2001.

[3] T. Ahmed, M. Coates, and A. Lakhina, "Multivariate online anomaly detection using kernel recursive least squares," in *Proc. IEEE Int. Conf. on Computer Communications (INFOCOM)*, Anchorage, AK, USA, May 2007.

[4] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least squares algorithm," *IEEE Trans. Signal Proc.*, vol. 52, no. 8, pp. 2275–2285, Aug. 2004.

[5] T. Ahmed, "Online anomaly detection using KDE," in *Proc. IEEE Global Communications Conf. (GLOBECOM)*, Honolulu, HI, USA, Nov. 2009.

[6] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," in *Proc. ACM SIGMETRICS*, New York, NY, USA, Jun. 2004.

[7] T. Ahmed, Sabrina Ahmed, Supriyo Ahmed, and M. Motiwala, "Real-time intruder detection in surveillance systems using adaptive kernel methods," in *Proc. IEEE Int. Conf. on Communications (ICC)*, Cape Town, South Africa, May 2010.

[8] T. Ahmed and R. Rahman, "Survey of anomaly detection algorithms: Towards self-learning networks," in *Security of Self-Organizing Networks: MANET, WSN, WMN, VANET*, A.-S. K. Pathan, Ed. Auerbach Publications, CRC Press, USA, Sep. 2010, pp. 65–89.

[9] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computat.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.

[10] C. Au, S. Skaff, and J. Clark, "Anomaly detection for video surveillance applications," in *Proc. IEEE Int. Conf. on Pattern Recognition (ICPR)*, Hong Kong, China, May 2006.

[11] T. Ahmed, B. Oreshkin, and M. Coates, "Machine learning approaches to network anomaly detection," in *Proc. ACM/USENIX Workshop on Tackling Computer Systems Problems with Machine Learning Techniques (SysML)*, Cambridge, MA, USA, Apr. 2007.

[12] C. Au, "Compression-based anomaly detection for video surveillance applications," Master's thesis, McGill University, Montreal, QC, Canada, Feb. 2006.

[13] T. Ahmed, X. Wei, S. Ahmed, and A.-S. K. Pathan, "Intruder detection in camera networks using the one-class neighbor machine," in *Proc. American Telecommunications Systems Management Association (ATSMA) Networking and Electronic Commerce Research Conf. (NAEC)*, Riva del Garda, Italy, Oct. 2011.

[14] ——, "Automated intruder detection from image sequences using minimum volume sets," *International Journal of Communication Networks and Information Security*, vol. 4, no. 1, pp. 11–17, Apr. 2012.

[15] X. Wei, T. Ahmed, M. Chen, and A.-S. K. Pathan, "PeerMate: A malicious peer detection algorithm for P2P Systems based on MSPCA," in *Proc. IEEE Int. Conf. on Computing, Networking and Communications (ICNC)*, Lahaina, HI, USA, Jan. 2012.

[16] X. Wei, J. Fan, M. Chen, T. Ahmed, and A.-S. K. Pathan, "SMART: A subspace based malicious peers detection algorithm for P2P systems," *International Journal of Communication Networks and Information Security*, vol. 5, no. 1, pp. 1–9, Apr. 2013.