A Prototype Application for Long-time Behavior Modeling and Abnormal Events Detection

Nicoletta Noceti and Francesca Odone
DIBRIS, Università degli Studi di Genova, via Dodecaneso 35, Genova, Italy

Keywords: Behaviour Modelling, Abnormal Events Detection, Temporal Series Clustering.

Abstract: In this work we present a prototype application for modelling common behaviours from long-time observations of a scene. The core of the system is based on the method proposed in (Noceti and Odone, 2012), an adaptive technique for profiling patterns of activities on temporal data – coupling a string-based representation and an unsupervised learning strategy – and detecting anomalies — i.e., dynamic events diverging with respect to the usual dynamics. We propose an engineered framework where the method is adopted to perform an online analysis over very long time intervals (weeks of activity). The behaviour models are updated to accommodate new patterns and cope with the physiological scene variations. We provide a thorough experimental assessment, to show the robustness of the application in capturing the evolution of the scene dynamics.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

1 INTRODUCTION

The essence of video-surveillance is to be able to analyse a scene for very long time frames and to automatically detect anomalous events. In contrast, researchers often refer to short benchmark video sequences when analysing the performances of their methods. For this reason it is rather difficult to estimate the robustness of a method with respect to the natural evolution of a scene along time. In spite of the considerable amount of work carried out until now (Johnson and Hogg, 1995; Bulpitt and Sumpter, 2000; Javed and Shah, 2004; Hu et al., 2007), real video-surveillance systems are still relying heavily on inputs from human operators. In practice, when a new system is installed, a configuration which highlights forbidden behaviours is mandatory. We are still missing automatic procedures able to *assist the user* in defining patterns of common behaviours, suggesting situations of potential danger.

This paper presents an engineered prototype of a monitoring application, the core of which is based on the theoretical model proposed in (Noceti and Odone, 2012). Such a model, which may be applied to different video analysis tasks, relies on the availability of a set of trajectories to describe meaninfull dynamic events occurring in a scene. Then, unsupervised learning is adopted to build models of common patterns of activities in the environment by exploiting a string-based representation and a clustering approach from long-time observations of the scene dynamics. As a side effect, the model provides the means to detect anomalous events.

Here we apply the model to a video-surveillance setting. Our goal is to produce an adaptive model of behaviours that are common in the observed scene. Dynamic events observed over a long temporal span are used to first initialise a model of common behaviours and then update it over time if it is needed. We discuss different strategies to update the behaviour models over time, to accomodate the effects of scene variations. Unlike in our previous works, the experimental procedure we adopt here is *online* and it incorporates new behaviour models when necessary.

Figure 1 shows a visual representation of the prototype monitoring system. The video-surveillance

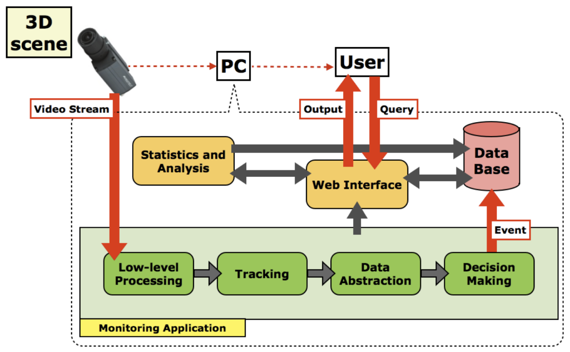


Figure 1: A visual representation of the prototype monitoring application.

Noceti, N. and Odone, F.
A Prototype Application for Long-time Behavior Modeling and Abnormal Events Detection.
DOI: 10.5220/0005723105970604
In *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2016) - Volume 4: VISAPP*, pages 597-604
ISBN: 978-989-758-175-5
Copyright © 2016 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

camera[1] is connected to a computer in which a devoted software organises the acquisitions and runs the monitoring application. The analysis in performed continuously over time.

When a new dynamic event is observed and classified, an event is generated and described with a set of tags – including type of event (normal event with information on the assigned class, or anomaly), starting and ending time, description, ... – which are collected on a data base. The final user can constantly monitor the system status via a web machine-user interface, where the intermediate output of each stage of the pipeline can be visualised. Moreover, the user is allowed to formulate queries to the data base to analyse the results and obtain statistics.

Previous Works. Profiling behaviours from temporal data is a research domain with a long history, where the benefit of adopting machine learning techniques has soon been observed. The problem has been tackled using supervised methods, as Support Vector Machines (Chen and Aggarwal, 2011) or Hidden Markov Models (F. Bashir and Schonfeld., 2007). However many applications– in particular within the video analysis domain– are characterised by the availability of huge sets of data but relatively few labeled ones; therefore an increasing interest has been posed to the unsupervised perspective. A rather complete account of event classification in an unsupervised setting is reported in (Morris and Trivedi, 2008); see also (Stauffer and Grimson, 2000; Hu et al., 2006).

More recently, there has been a renewed attention on the problem of detecting abnormal events. We cite for instance the work in (Cheng et al., 2015), where a local and global anomalies are detected using a regression model on Spatio-Temporal Interesting Points (STIPs), (Ren et al., 2015) that employs Dictionary Learning to build models of common activities, or again (Xu et al., 2015), which is built on top of the by now popular Deep Neural Networks.

2 MONITORING APPLICATION

We address the problem of modelling behaviours in a setting where people are observed as a whole and their dynamic can be described by a trajectory of temporal observations. A behaviour can then be formalised as a set of dynamic events coherent with respect to a certain metric (e.g. *going from a point A to point B* or *enter the region C*). Depending on the available information on the trajectories, more subtle properties can be enhanced, e.g. if velocity is considered, then the behaviour *going from A to B* can be further divided into *going from A to B by walking* or *by running*.
In the following we describe our monitoring application. On a first period, we simply collect dynamic events representations, and then run the training stage in batch to obtain an initial guess on the patterns of common activity in the scene. The core of this step is the method we proposed in (Noceti and Odone, 2012), summarised in Sec. 2.1 and 2.2 (we refer to the original paper for further details). Then, the online test analysis starts. If necessary, the behaviours models are updated to address the problem of adapting to the scene changes. We discuss this point if Sec. 2.3.

2.1 Modelling Common Activities in a Scene

The method we adopt consists of different steps. A low-level processing collects trajectories over time, which in a second step are mapped into strings representations. Finally, groups of coherent strings are detected with clustering.

2.1.1 Low-level Video Processing

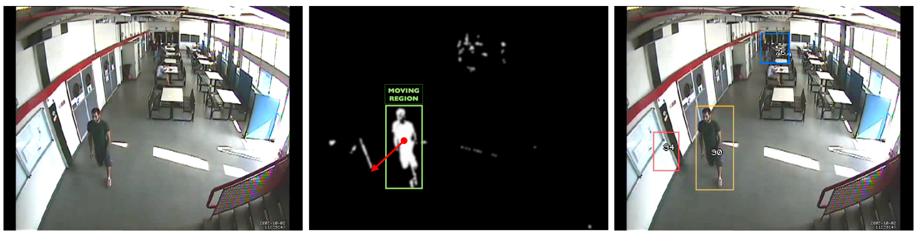
Our low-level processing (Noceti et al., 2009) aims first at performing a motion-based image segmentation to detect moving regions (see Fig. 2(b)), that in our setting can be associated with a single person or a small group of people (see Fig. 2(c)). At a given time instant t, each moving object i in the scene is described according to a vector of feature $x_i^t = [\mathbf{P}_i^t, S_i^t, M_i^t, D_i^t] \in \mathbb{R}^5$, where $\mathbf{P}$ denotes the object position in the image plane, S its size, M and D the magnitude and orientation of the object motion.

The tracking procedure correlates such vectors over time (Fig. 2(c)), obtaining N spatio-temporal trajectories that are collected on a training set of temporal series $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$.

2.1.2 String-based Trajectory Representation

We map each temporal trajectory into a new representation based on strings – formally, a concatenation of symbols from a finite *alphabet* $\mathcal{A}$. The alphabet can be obtained by partitioning the space of all vectors x_i^t disregarding the temporal correlation: $\tilde{\mathbf{X}} = \{\{x_i^t\}_{t=1}^{k_i}\}_{i=1}^N$, where k_i refers to the length of the i-th trajectory. We adopted the method proposed in (Noceti et al., 2011), where the benefit of adopting a strategy guided by the available data, as opposed to

[1]The video surveillance setup has been obtained within a technology transfer program with Imavis S.r.l., http://www.imavis.com.



(a) (b) (c)

Figure 2: The low-level processing of the monitoring system, based on (Noceti et al., 2009): each frame (left) is first segmented to detect moving regions (center). Then each region is described with a feature vector used in the tracking process to assign and maintain the identity of the subject or group of people (right).

a manual or a regular grid has been discussed. The partition method relies on clustering data in $\tilde{\mathbf{X}}$ with a recursive implementation of *spectral clustering* (Shi and Malik, 2000). A nice property is that it does not require to fix a priori the number clusters, while still allows to influence the granularity of the solution with appropriate parameter tuning.

As for the similarity measure, we adopt the function proposed in (Noceti et al., 2011), based on a convex combination of kernel-based similarity functions computed on sub-sets of coherent features:

$$\begin{aligned} G(x_1,x_2,\mathbf{w},\theta) = & w_{\mathbf{P}} G_{\mathbf{P}}(\mathbf{P}_1,\mathbf{P}_2,\sigma_{\mathbf{P}})+ \\ & w_S G_S(S_1,S_2,\sigma_S)+ \\ & w_M G_M(M_1,M_2,\sigma_M)+ \\ & w_D G_D(D_1,D_2,\sigma_D), \end{aligned} \tag{1}$$

where $\mathbf{w} = \{w_{\mathbf{P}}, w_S, w_M, w_D\}$ are the feature weights, while $\{G_{\mathbf{P}}, G_S, G_M, G_D\}$ are Gaussians with standard deviations $\theta = \{\sigma_{\mathbf{P}}, \sigma_S, \sigma_M, \sigma_D\}$ and zero means.

Once that the clustering has been performed, a set of states composing the features partition is available. The alphabet can be finally obtained by associating a symbol with each state.

At this point, a temporal series $\mathbf{x}_i$ can be translated into a string s. Each element $x_i^t \in \mathbb{R}^5$ is compared (using again Eq. 1) with the centroid of the states. Then, the element is represented in the string by the label associated with the closest state. We now have obtained the set $\mathbf{S} = \{s_i\}_{i=1}^N$, composed by the string representations of the original training set $\mathbf{X}$.

2.1.3 Behavior Modeling using Strings Clustering

The method we adopt to discover patterns of common activities includes three steps: (i) strings clustering, (ii) candidates selection, and (iii) clusters refinement.

Step (i) is again based on Spectral Clustering, which is run on the strings representation with the aim of detecting groups of coherent strings. To evaluate the similarity between strings we adopt the *P-spectrum kernel* (see (Taylor and Cristianini, 2004)). The peculiarities of our data reside on the transitions between states, thus we set $P = 2$ and discard the replicated symbols from each strings before applying spectral clustering. We refer the interested reader to (Noceti and Odone, 2012) for the theoretical justifications.

We now have a string partition $\mathcal{B}$, defining groups of coherent temporal events, including the strings clusters $\{S_1,\ldots,S_{|\mathcal{B}|}\}$. To represent each of them in a compact way we select a *cluster candidate* with a voting strategy (step (ii)). In what follows, we will refer to the candidate of strings cluster S_i with $\hat{s}_i$. Intuitively, the candidate string can be interpreted as an "average" string within a cluster. Only the votes corresponding to a high similarity are considered: notice that this is equivalent to discard outliers from clusters before candidates selection, obtaining a method which is tolerant to the presence of noisy data.

Given the strings partition $\mathcal{B}$, a refinement stage (step (iii)) is applied to improve the quality of clusters, given that Spectral Clustering may produce over-segmented results on real, noisy data. To this purpose, first low-quality clusters (groups with either a low intra-class similarity or a small cardinality) are removed. Then, clusters with very similar candidates are merged.

2.2 Model Selection with Loose Prior Information

The choice of parameters influences the obtained models, thus it is worth discussing how to appropriately assign their a value. In our case, they include the weights $\mathbf{w}$ (see Eq. 1) and the parameters used for Spectral Clustering to control the granularity of the solution (the so-called *cut thresholds*, referred to as τ_A for the alphabet and τ_S for the strings partition).
When a ground truth is available, then the best clustering solution may be chosen by solving a 1-to-1 assignment problem between estimated clusters and expected clusters. However, in many real circumstances

with the availability of huge amounts of data, obtaining a manual and full annotation might be painful and generate very subjective views of the problem.

We adopt instead a viable strategy which involves the *annotation of the observed environment*, rather than of the dynamic events. To goal is to manually identify regions of the scene which are characterised by some level of relevance (as in presence of doors of coffee urns). Figure 5 shows an example of an annotation for the scene used in our experiments. We refer to this annotation as a *loose annotation*, in that it provides a partial annotation of the dynamic events, which are grouped according to starting point (source region) and final point (sink region). It goes that only spatial properties influence the annotation, and as a consequence it may generate heterogeneous groups: trajectories sharing source and sink regions may correspond to different classes of objects or motion type – e.g. the event *going directly from the source to the sink* will be grouped together with events crossing other intermediate regions.

For these reasons, a 1-to-N mapping is more appropriate to our case: an annotated cluster is allowed to be associated with multiple estimated clusters to avoid high penalties for over-segmented but reasonable results (Brox and Malik, 2010). In fact, since the ground truth only relies on spatial properties, an annotated cluster may be split in more than one estimated clusters because other properties are captured (e.g. people walking vs people running, people vs cars, ...). Hence, an estimated cluster is always associated with the ground truth cluster with highest intersection.

We compute the Correct Clustering Rate (CCR) (Morris and Trivedi, 2009) to evaluate the goodness of a given clusters mapping. Consequently, model selection simply implies to select the parameters that maximize such value.

2.3 Evolving the Models over Time

After the training stage the system starts to monitor the scenario analysing the new observed events. The training stage described so far is *batch*, thus models are maintained as they have been estimated during training. Instead, it goes that the models should be able to automatically update to cope with the temporal variations in the scene.

The models might be updated at fixed intervals – e.g. each night or in the week-end to avoid system overloading during daytime analysis – or when the performances significantly decrease. In an unsupervised setting there is no direct way to evaluate such deterioration, but we may consider a constantly increasing amount of detected anomalies as an indication of the fact models might be becoming obsolete. In our scenario, if the number of dynamic events associated with known models is very low then it might be the case to replicate the *entire training pipeline*, since some features may had acquired importance with respect to others and thus the alphabet to build the string-based representation may be varied. However, usually we can assume the alphabet is still valid, while the patterns of activities may be more affected by variations in the scene.

Starting from the strings in the original training set, we refer to as $\mathbf{S}^0 = \{\mathbf{s}_i\}_{i=0}^{N_S^0}$, from which the partition $\mathbf{C}^0 = \{C_i^0\}_{i=1}^{N_C^0}$ of N_C^0 clusters have been estimated and represented via the set of candidates $\mathbf{c}^0 = \{c_i^0\}_{i=1}^{N_C^0}$, we first discard the strings in $\mathbf{S}^0$ whose similarity with their candidates is below a threshold. Then we add the new observed strings $\mathbf{S}^{\Delta t} = \{\mathbf{s}_j\}_{j=0}^{N_S^{\Delta t}}$ over a temporal window of Δt frames: the interval under consideration should cover the period occurred since the last update (or the training stage). If the number of strings considered in this way is too high the clustering steps becomes computationally unfeasible, and thus a random sampling can be applied. Since the new training stage involves the strings clustering step only, the model selection procedure is limited to the choice of the corresponding cut threshold, τ_S.

3 LEARNING BEHAVIORS: LONG-TERM EXPERIMENTS

In this section we discuss the performances of our approach to behaviour modeling over the long time period. The monitored environment, shown in Fig. 5, is one of the main hall of our department. The environment consists in an indoor open space where a good amount of dynamic events occur during daytime. Only people are supposed to be moving in the scene, in which the complexity depends on several factors, as level of crowd or illumination changes.

The analysis is performed over a period of 7 weeks, characterised by a different amount of dynamic activity. The storyline of the scene is depicted in Fig. 3. On the first week, where normal activity can be observed, we simply collect data and train our initial models. On the second and third weeks, characterised by a normal activity as well, we test our model. We expect a few anomalies to be identified in this period. On the fourth week a special event occurs (the department was open to high school students), and some structural variations are applied to the scene during the week (tables and bulkheads are temporary rearranged, camera position slightly changes). As a consequence we expect that both the amount of observed dynamic events and detected anomalies increase. To follow, there are two weeks of holidays, in which very few dynamics should be observed, to go back to a normal activity on the last week of analysis. In this last week exam sessions took place, with a consequent significant activity in the area of the table (due to students studying).

We now detail the temporal analysis.

3.1 Training the Initial Models

We collected data for training the initial models over the first week, gathering a training set of 1203 dynamic events, loosely annotated according to the 9 source-sink regions reported in Fig. 5.

The model selection procedure selected as best parameters $W_{\mathbf{P}} = 0.5$, $W_S = 0.1$, $W_M = 0.2$ and $W_D = 0.2$, $\tau_A = 0.85$ and $\tau_S = 0.75$ (the values of the sigma's were $\sigma_{\mathbf{P}} = 70.278$, $\sigma_S = 5631.8$, $\sigma_M = 1.278$ and $\sigma_D = 0.597$). The model corresponds to a clustering including the 10 behaviours reported in Fig. 4 and represented with arrows in Fig. 5. It is easy to observe that the identified patterns can be associated with rather common and intuitive activities occurring in the scene.

Notice how the behaviours pairs 2-5 and 6-8 show a significant spatial overlap. For the latter, an inspection of the other features indicates the difference between them which has been embedded in the clustering solution. Indeed, the distributions of the area feature of all the observations of trajectories included in the clusters reveal rather different average values (~ 1500 for pattern 6, ~ 3000 for pattern 8) that may refer to single subjects and small groups, respectively. The separation of the pair 2-5 is likely to be due to an over-segmentation of the data. However, trajectories included in pattern 2 show a higher spatial compactness, which is also reflected in the distribution of the direction features (characterised by a very low standard deviation).

3.2 Run-time Analysis

In this section we discuss the run-time analysis: given the models estimated after the training stage the system acquires new observations, describes the dynamic events according to the scheme representation followed during training and adopting the parameters selected as best performing, and finally tries and associate the new datum with one of the known models. The association follows a *Nearest Neighbour* strategy: the new datum, represented as string, is compared with all the known models, i.e. their candidates. If no candidate has a high similarity (above a threshold, 0.6 in our experiments) with the new string, then the new event is considered anomalous.

Let us first have a look at the number of events observed during the period of analysis: the plot, reported in Fig. 6(a) shows that the trends of each week present a rather similar shape, with peaks in the middle (Wednesday's). Also, the weeks just before and after the holidays show a higher amount of activity.

After the first week of training, we performed the classification analysis over the remaining weeks, with the aim of discriminating between normal dynamics and anomalies. The analysis is extensively discussed on the next sections.

3.2.1 With Fixed Models

As a baseline, we adopted the models estimated during training over the whole period of analysis (no updating).

Figure 6(b) reports the number of anomalous events detected over the weeks, included the statistics on the first week (training) that serve as a reference value. The fraction of anomalies detected in the 4th week is higher than in the previous days, and even higher after holidays. This is justified by the special event scheduled during the 4th week, in which the activities diverged with respect to the common dynamics of the environment due to both the presence of outer people and the variation on the scene structure. The latter also justifies the presence of many anomalies after the holidays weeks. Indeed, the average percentage of anomalies over the weeks (with the exception of the holidays interval, in which only a few dynamic events have been observed) shows a growth over time (W2: 0.12, W3: 0.15, W4: 0.25, W7: 0.27), suggesting that the initial models may be becoming more and more obsolete. Notice that although the amount of measured dynamic events is significantly higher on the last week, the fraction of anomalous events is comparable to the corresponding statistics of week 4, where scene variations have been applied.

This suggests the benefit of updating the initial models with new observations from the environment. Possible solutions are discussed in what follows.

3.2.2 With Model Updating

The choice regarding *when* updating the system can be done following different criteria, we considered 3 different strategies. The first strategy (Fig. 7(a)) relies on updating the models at fixed time intervals, and

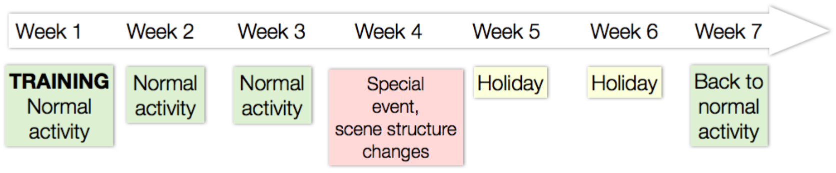
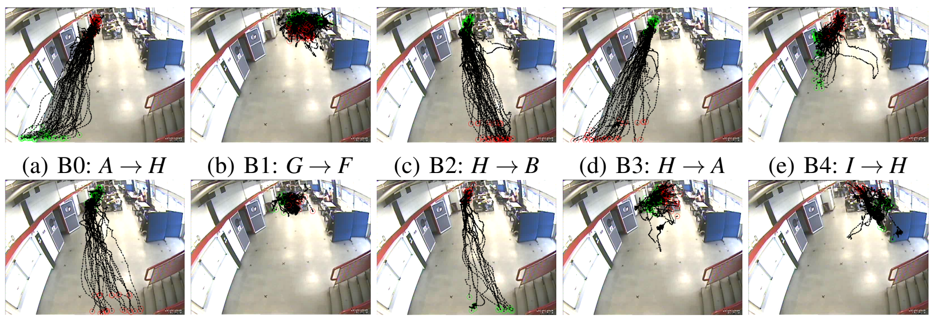


Figure 3: Storyline of the long-time period of analysis of our prototype application.



(a) B0: $A \rightarrow H$ (b) B1: $G \rightarrow F$ (c) B2: $H \rightarrow B$ (d) B3: $H \rightarrow A$ (e) B4: $I \rightarrow H$

(f) B5: $H \rightarrow B$ (g) B6: $H \rightarrow F$ (h) B7: $B \rightarrow H$ (i) B8: $H \rightarrow F/G$ (j) B9: $E \rightarrow G$

Figure 4: Behaviors learned on the training phase. Source (in green) and sink (in red) locations are also reported.

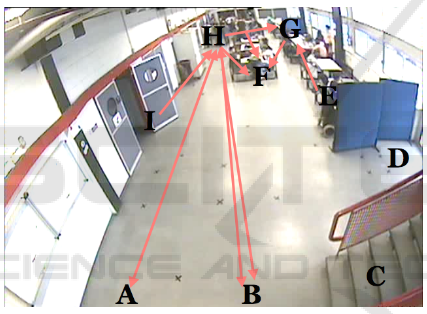
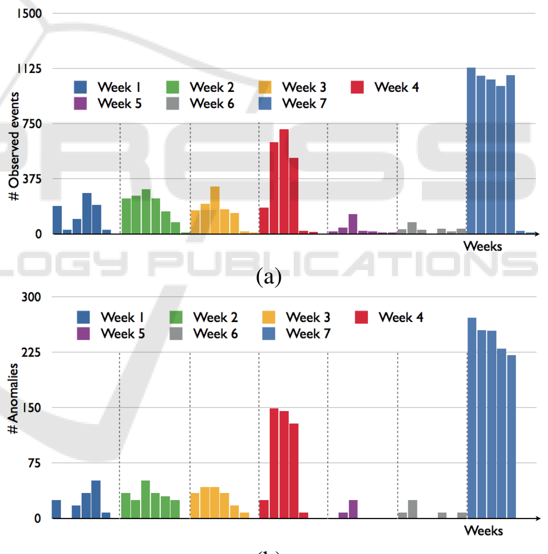


Figure 5: Patterns of activity detected with the training stage.



(b)

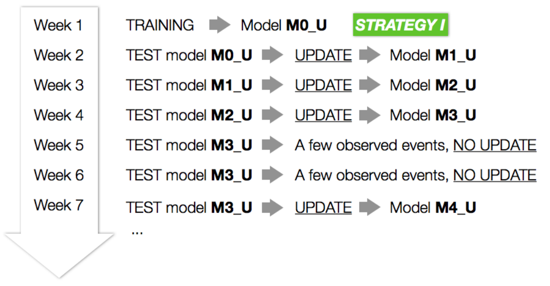
Figure 6: The amount of observed events (above) and of anomalies detected if no updating rule is applied (below) during the whole period of analysis.

more specifically at the end of each week, during the week-end, where the computational load of the system is significantly lower due to the lack of dynamics in the scene. The step is performed only when the amount of observed events is significant. Thus, at the end of training we have at disposal the initial unsupervised model M0_U, which is updated to model M1_U at the end of the second week. During holidays no update takes place due to the scarcity of dynamics.

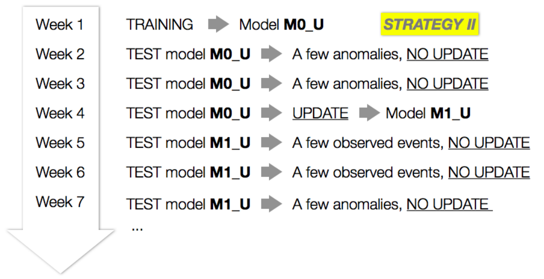
An alternative to avoid the computational weight of updating procedures which are not strictly necessary is to update the models when the performances decrease too much. In our implementation (Fig. 7(b)) we evaluate the percentage of anomalies with respect to the total number of dynamic events, to trigger a models update when the amount goes above a given threshold (here 20%). In the specific experimental scenario, this corresponds to a unique models update, performed at the end of week 4.

A third strategy of updating relies on the use of a different trajectory classifier to be adopted when the models become stable. Figure 7(c) shows a possible implementation leveraging on the use of traditional *Support Vector Machine* (SVM) equipped with the P-Spectrum kernel. The training stage considers the same training set as in the unsupervised counterpart, with strings labels according to the annotation induced by the clustering result selected as optimal (best fitting the loose annotation). A k-fold cross validation (k=5) has been used to select the parameters

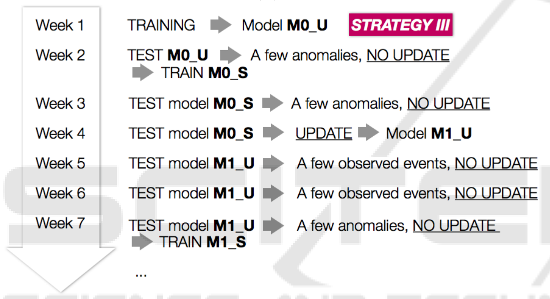
SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS



(a)



(b)



(c)

Figure 7: Different strategies for models evolution.

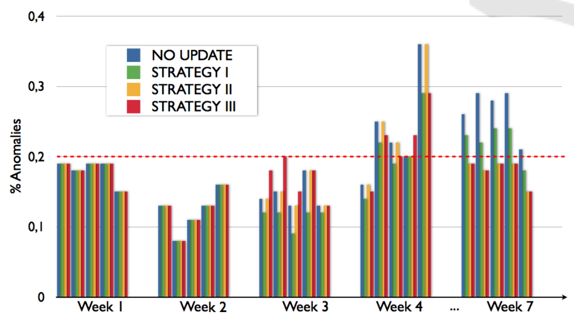


Figure 8: Anomalies detected over time comparing the 3 strategies for models updating.

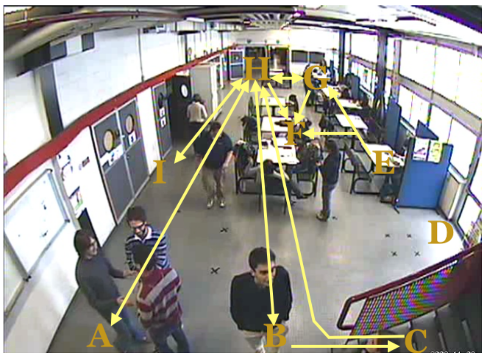
of SVM's. In our experimental setting, models are considered as stable after the second week, so the initial unsupervised models are employed for training the supervised model M0_S, which becomes now the active model adopted for the classification of the new observed data. At the end of week 4, where a higher amount of anomalies has been detected, a new training is performed to estimate the updated unsupervised models, M1_U, which is converted to the supervised counterpart only in presence of an appropriate amount of dynamics and a low rate of anomalies. This corresponds to the end of week 7.

Figure 8 shows the trend of the anomalies percentage over the weeks of analysis (with the exception of holidays) comparing the different update strategies – we will refer to as NU (no update), S1 (strategy 1), S2 (strategy 2) and S3 (strategy 3). On the first two weeks, the performances are equal because the behavior models correspond to the original cluster in all the three strategies. On week 3, the first model update at the end of week 2 causes a gain in performances to S1. The update triggered at the end of the special event (week 4) is not sufficient to S1 for performing appropriately on the last week. Instead, the update performed with S2 (at the end of week 4 as well) guarantees a more stable solution thanks to the higher amount of dynamic events adopted for the new training stage (subsampled from the observations of weeks 2, 3 and 4). This reflects on the fact that the amount of anomalies detected on week 7 goes back to the accepted level (below the threshold, in red in the picture). The supervised classifier employed in S3 seems to have performances comparable to the unsupervised counterpart. This suggests that even in the unsupervised case, where the labels are not explicitly employed in the modeling stage, the prior information is appropriately exploited to build models which are robust to noise.
We conclude with some final remarks on the patterns detected after the models updating with S2 at the end of week 4 (Fig. 9(a)) and examples of clusters so included in the models while classified as anomalies by the original learnt patterns (see Fig. 9, below). As expected, a richer description of the scene dynamics is achieved.

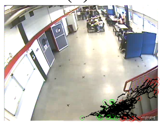
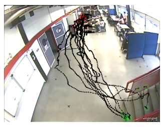
4 CONCLUSION

In this paper we presented an engineered prototype of a monitoring application, adopting the method for behaviour modelling proposed in (Noceti and Odone, 2012). The application is able to collect a low-level description of a dynamic event in terms of a trajectory, which is then turned into a string-based representation that captures the peculiarities of the data. Then, common patterns of activities are detected adopting unsupervised learning – and clustering, in particular – on large sets of dynamic events. During the run-time analysis, a new dynamic event is described according to the same procedure, and classified as instance of one of the known behaviour, or as an event which di-

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS



(a)

(b) $H \rightarrow I$ (c) $B \rightarrow C$ (d) $C \rightarrow H$

Figure 9: Above, patterns of activity detected after the models updating step. Below, examples of obtained clusters.

verges with respect to the normality, thus an anomaly.

We discussed the use of three different strategies to update the models, starting from an initial configuration, and evaluated the monitoring application on a period of seven weeks of analysis. The results show the robustness of the method over time, which provides a tolerance with respect to variations in the environment. The update of the models allows to accommodate for new patterns which are initially associated with the anomaly class but then turned into a common activity due to the persistence over time.

REFERENCES

Brox, T. and Malik, J. (2010). Object segmentation by long term analysis of point trajectories. In *ECCV*, pages 282–295.

Bulpitt, A. and Sumpter, N. (2000). Learning spatio-temporal patterns for predicting object behaviour. *IMAVIS*, 18(9):697–704.

Chen, C. and Aggarwal, J. (2011). Modeling human activities as speech. In *CVPR*, pages 3425–3432.

Cheng, K.-W., Chen, Y.-T., and Fang, W.-H. (2015). Video anomaly detection and localization using hierarchical feature representation and gaussian process regression. In *CVPR*, pages 2909–2917.

F. Bashir, A. K. and Schonfeld., D. (2007). Object trajectory-based activity classification and recognition using hidden markov model. *IP*, 16(7):1912–1919.

Hu, W., Xiao, X., Fu, Z., Xie, D., Tan, T., and Maybank, S. (2006). A system for learning statistical motion patterns. *PAMI*, 28(9):1450–1464.

Hu, W., Xie, D., Fu, Z., Zeng, W., and Maybank, S. (2007). Semantic-based surveillance video retrieval. *IP*, 16(4):1168–1181.

Javed, I. J. O. and Shah, M. (2004). Multi feature path modeling for video-surveillance. In *ICPR*, pages 716–719.

Johnson, N. and Hogg, D. (1995). Learning the distribution of object trajectories for event recognition. In *BMVC*, volume 2, pages 583–592.

Morris, B. and Trivedi, M. (2008). A survey of vision-based trajectory learning and analysis for surveillance. *Circ. and Sys. for Video Tech.*, 18(8):1114–1127.

Morris, B. and Trivedi, M. M. (2009). Learning trajectory patterns by clustering: Experimental studies and comparative evaluation. In *CVPR*, pages 312–319.

Noceti, N., Destrero, A., Lovato, A., and Odone, F. (2009). Combined motion and appearance models for robust object tracking in real-time. In *AVSS*, pages 412–419.

Noceti, N. and Odone, F. (2012). Learning common behaviors from large sets of unlabeled temporal series. *Image and Vision Computing*, 30(11):875–895.

Noceti, N., Santoro, M., and Odone, F. (2011). Learning behavioral patterns of time series for video-surveillance. In *Learning Behavioral Patterns of Time Series for Video-Surveillance (Springer)*, pages 275–304. Springer-London.

Ren, H., Liu, W., Olsen, S., and Moeslund, T. (2015). Unsupervised behaviour-specific dictionary learning in abnormal event detection. In *BMVC*.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *Trans. on PAMI*, 22(8):888–905.

Stauffer, C. and Grimson, E. (2000). Learning patterns of activity using real-time tracking. *PAMI*, 22(8):747–757.

Taylor, J. S. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Xu, D., Ricci, E., Yan, Y., Song, J., and Sebe, N. (2015). Learning deep representations of appearance and motion for anomalous event detection. In *BMVC*.