International Journal of Computer Vision (2019) 127:993–1011
https://doi.org/10.1007/s11263-018-1129-8

Fast Abnormal Event Detection

Cewu Lu[1] · Jianping Shi[2] · Weiming Wang[3] · Jiaya Jia[4]

Received: 22 February 2016 / Accepted: 27 July 2018 / Published online: 1 December 2018
© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract
Fast abnormal event detection meets the growing demand to process an enormous number of surveillance videos. Based on the inherent redundancy of video structures, we propose an efficient sparse combination learning framework with both batch and online solvers. It achieves decent performance in the detection phase without compromising result quality. The extremely fast execution speed is guaranteed owing to the fact that our method effectively turns the original complicated problem into a few small-scale least square optimizations. Our method reaches high detection rates on benchmark datasets at a speed of 1000–1200 frames per second on average when computing on an ordinary single core desktop PC using MATLAB.

Keywords Abnormal event · Realtime detection · Event detection · Video analysis

1 Introduction

With the increasing demand of security, surveillance cameras are commonly deployed. Detecting abnormal events is a critical task based on what cameras capture, which is traditionally labor-intensive and requires non-stop human attention. What makes this interminable and boring process worse is that concentration levels are hard to maintain so that infrequent abnormalities may be missed. This predicament catalyzes important research in computer vision, aiming to find abnormal events automatically (Jianga et al. 2011; Adam et al. 2008; Mahadevan et al. 2010; Saligrama and Chen 2012a; Zhang et al. 2005; Kratz and Nishino 2009; Zhao et al. 2011; Cong et al. 2011).

It is not a typical classification problem due to the difficulty to list all possible negative samples. Research in this area commonly follows the line that normal patterns are first learned from training videos, and are then used to detect events deviated from this representation. Recently, sparse representation (Lu et al. 2013b; Shi et al. 2011) has attracted much attention and sparsity-based abnormality detection models (Zhao et al. 2011; Cong et al. 2011) achieved state-of-the-art performance reported in many datasets.

Although realtime processing is a key criterion to a practically employable system given continuously captured videos, most sparsity-based methods cannot perform fast enough. The major obstruction to high efficiency is the inherently intensive computation to build the sparse representation. Note a slow abnormal event detection process could delay alarm and postpone response to special events.

In what follows, we provide brief analysis about this issue with respect to the general sparsity strategies and present our new framework with an effective representation. It fits the structure of surveillance videos and leads to an extremely cheap testing cost.

Communicated by Kristen Grauman.

✉ Cewu Lu
lucewu06@hotmail.com

✉ Weiming Wang
wangweiming@sjtu.edu.cn

1 Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China

2 Sensetime Group Limited, Beijing, China

3 School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China

4 Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong, China

1.1 Sparsity Based Abnormality Detection

Sparsity is a general constraint (Zhao et al. 2011; Cong et al. 2011) to model normal event patterns as a linear combination of a set of basis atoms. We analyze abnormality detection in one local region to show that this process is computationally expensive by nature.

Given training features $[\mathbf{x}_1, \ldots, \mathbf{x}_n]$ extracted from the history video sequence in a region, a normal pattern dictionary $\mathbf{D} \in \mathbb{R}^{p\times q}$ is learned with a sparsity prior, where p and

q are basis vector dimension and atom number respectively. In the testing phase for a new feature $\mathbf{x}$, we reconstruct it by sparsely combining elements in $\mathbf{D}$, expressed as

$$\min_{\boldsymbol{\beta}} \|\mathbf{x} - \mathbf{D}\boldsymbol{\beta}\|_2^2 \quad \text{s.t.} \quad \|\boldsymbol{\beta}\|_0 \leq s \tag{1}$$

where $\boldsymbol{\beta} \in \mathbb{R}^{q\times 1}$ contains sparse coefficients. $\|\mathbf{x} - \mathbf{D}\boldsymbol{\beta}\|_2^2$ is the data fitting term; $\|\boldsymbol{\beta}\|_0$ is the sparsity regularization term; and s ($\ll q$) is a parameter to control sparsity. With this representation, an abnormal pattern can be naturally defined as one with large error resulted from $\|\mathbf{x}-\mathbf{D}\boldsymbol{\beta}\|_2^2$. Previous work verified that this form can lead to high detection accuracy.

1.1.1 Efficiency Problem

A high testing cost is inevitable when adopting Eq. (1), which aims to find the suitable basis vectors (with scale s) from the dictionary (with scale q) to represent testing data $\mathbf{x}$. The search space is quite large, as $\binom{q}{s}$ different combinations exist. Although much effort has been put to reducing the dictionary size (Cong et al. 2011) and adopting fast sparse coding solvers (Zhao et al. 2011), in general, several seconds are needed to process a frame as reported in prior papers.

The efficiency problem is thus critical to address before this type of methods can be deployed practically. A realtime process needs to be 100 times faster than the current fastest sparsity-based methods, which is difficult without tremendous hardware advancement. We tackle this problem from an algorithm perspective. Our method yields decent performance and naturally accelerates sparse coding by 2000+ times even using MATLAB implementation.

1.2 Our Contribution

We propose *sparse combination learning* (SCL) for detection. With high structure redundancy in surveillance videos, instead of coding sparsity by finding an s basis combination from $\mathbf{D}$ in Eq. (1), we code it directly as a set of possible combinations of basis vectors. Each combination here corresponds to a set with s dictionary bases in Eq. (1). With this change, other than searching for s bases from q of them for each testing feature, we only need to find the most suitable combination by evaluating the least square error. The testing framework is shown in Fig. 1.

This framework is efficient since only small-scale least square optimization is required in detection with simple matrix projection. In our experiments, testing is on a small number of combinations, each takes 10^{-6}–10^{-7} second in MATLAB.

The effectiveness of our approach is well guaranteed by the inherent sparsity constraint on the combination size. Compared to original sparse coding, our model is more faithful to the input data. When freely selecting s basis vectors from a total of q vectors by Eq. (1), the reconstructed structure could much deviate from input due to the large freedom. But in our trained combinations, it is unlikely to happen, since each combination finds its corresponding input data, better constraining reconstruction quality. Our method therefore is robust to distinguish between normal and abnormal patterns.

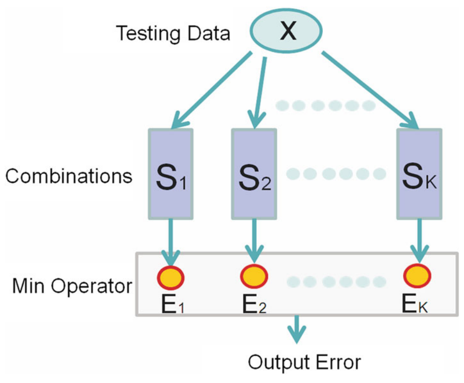

Fig. 1 Our testing architecture. $\mathbf{X}$ denotes testing data. $\{\mathbf{S}_1, \ldots, \mathbf{S}_K\}$ are learned combinations, with each $\mathbf{S}_i \in \mathbb{R}^{p\times s}(s \ll p)$. E_i is the corresponding least square reconstruction error. The final error is the minimum among all combinations

We have verified our model on a large set of surveillance videos in Sect. 6.2. We also benchmark it on existing datasets for abnormal event detection. The detection speed is further improved by our optimized implementation compared to our conference version (Lu et al. 2013a). It reaches 1000–1200 FPS using a desktop with a 3.4GHz CPU and 8G memory in MATLAB 2012.

This manuscript extends its conference version (Lu et al. 2013a) with the following major differences. (1) We extend our sparse combination learning into a birth-and-death combination online solver to handle large-scale and dynamic data. (2) We double the size of our Avenue dataset, and label the spatial locations of the abnormal event for accurate evaluation. We further propose a large scale dataset NYDP with 7 h training data to evaluate the performance under large scale data. (3) We conduct extensive experiments.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 introduces the sparse combination learning framework. An online extension for sparse combination learning is proposed in Sect. 4. In Sect. 5, we present our fast testing scheme. Section 6 shows results on several benchmark datasets.

2 Related Work

2.1 Abnormal Event Detection

Abnormal detection in surveillance refers to the problem of finding unusual patterns in videos. A complete survey for anomaly detection is in Chandola et al. (2009). Usually, abnormal event detection can be casted into categories.

Early work (Shet et al. 2006) explicitly defined the logic to find anomaly. However, this method cannot be generalized to complex scenes. Following work aims to build different models based on different video features. Three major categories are included.

One sort of models utilize the normal event as a multivariate Gaussian mixture model (GMM) (Basharat et al. 2008; Wu et al. 2010; Mahadevan et al. 2010; Li et al. 2014; Shi et al. 2010), which includes the clustering based methods as a special case (Jiang et al. 2007, 2008). These methods utilize each central signal to represent a group of normal patterns. Thus, they may involve a large amount of centers to represent all normal cases. If the number of training data is limited, the normal data space cannot be fully supported. Also, the reconstruction ability by using a single central signal for a group of normal signals is restrictive. Thus, the resulting detection accuracy could be reduced due to errors or inappropriate clustering. Recently, a model using Gaussian process regression (Cheng et al. 2015) was proposed. Interaction from training videos is modeled as seeking frequent geometric relation of nearby sparse interest points.

Another set of models fit video cubes into graph models in Zhang et al. (2005), Kratz and Nishino (2009), Kim and Grauman (2009), Benezeth et al. (2009), Wang et al. (2007), Mehran et al. (2009), Cui et al. (2011), Antic and Ommer (2011), Jager et al. (2008) and Pruteanu-Malinici and Carin (2008), which utilize co-occurrence patterns. Among these methods, normal patterns were fitted in a space-time Markov random field (MRF) in Zhang et al. (2005), Kratz and Nishino (2009), Kim and Grauman (2009), Jouseok and Kyoungmu (2012) and Benezeth et al. (2009). Nearly all MRF-based models use the cuboid window strategy, which may lose important information since meaningful features are separated into regions.

Topic models such as Latent Dirichlet Allocation (LDA), can also be adopted to detect anomaly (Wang et al. 2007; Mehran et al. 2009). However, this model would face difficulties in choosing the number of topics, and losing spatial information in video sequences.

There is work focusing on histogram representation. For example in Kaltsa et al. (2015), based on swarm theory, histograms of oriented swarms were proposed to describe each scene.

Xu et al. (2015) introduced deep learning in abnormal event detection. An appearance and motion DeepNet (AMDN) was presented, which can automatically learn feature representation for normal patterns.

Recently, sparsity based models (Lu et al. 2013b; Shi et al. 2011) have gained success in abnormal event detection (Zhao et al. 2011; Cong et al. 2011). They draw intrinsic components, i.e. dictionary atoms, to represent usual events. These methods can be easily generalized to online versions. The problem of sparse coding is that the testing process is time consuming, which may not satisfy real time detection tasks.

In short, several of these methods do not meet the realtime processing requirement for abnormal event detection. In this paper, we aim at high computation speed and propose a very fast solution with reasonable accuracy.

Previous work (Basharat et al. 2008; Mahadevan et al. 2010; Kim and Grauman 2009) also employed mixture of probabilistic principal component model (MPPCA) to summarize normal events. The normal patterns are mapped into multi-space in a probabilistic manner. If testing data is far away from the space, it is abnormal. Assumptions about data (e.g. Gaussian distributions for normal patterns) are required in these methods.

2.2 Subspace Clustering

Our proposed sparse combinations learning framework is related with subspace clustering. The early work follows Random Sample Consensus (RANSAC) (Fischler and Bolles 1981) where subspace is fitted by data randomly chosen from a pool. When there are many different combinations, the number of trials to find points in the subspace grows exponentially. Generalized principal component analysis (GPCA) (Vidal et al. 2005; Ma et al. 2008) seeks a polynomial whose gradient at a point is normal to the subspace. Thus, subspace clustering becomes fitting the polynomial. However, it is sensitive to noise and outliers. Complexity is also high.

In Elhamifar and Vidal (2013) and Elhamifar and Vidal (2009), sparse subspace clustering (SSC) was proposed. This set of methods are based on the fact that each point has a sparse reconstruction representation with regard to the dictionary formed by other data points. ℓ_1 sparse coding was thus adopted. This method is good at handling affine subspaces. It needs to have rough estimation of the number of subspaces. For abnormal event detection, this number is hard to know in prior and may vary in different cases.

Our approach can be regarded as enhancement of sparse subspace clustering (Elhamifar and Vidal 2009) with the major difference on the working scheme. The relationship between sparse subspace clustering (Elhamifar and Vidal 2009) and our method is similar to that between K-means and hierarchical clustering (Trevor et al. 2001). Specifically, for real data that is with noise, subspace clustering (Elhamifar and Vidal 2009) takes the number of clusters k as known or fixed beforehand, like K-means. In video abnormality detection applications, it is however difficult to know the optimal number of bases in prior. Our approach utilizes the allowed representation error to build combinations, where the error upper bound is explicitly implemented with clear statistical meaning. There is no need to define the cluster size in this method. In addition, our training cost is much cheaper.

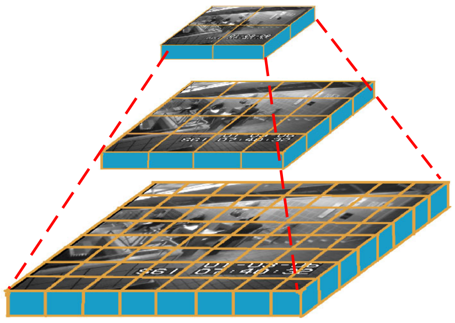

Fig. 2 Pyramid region architecture. A frame is resized into 3 different scales. In each scale the frame is partitioned into several regions

We only need several minutes to training 10K-sample data, while sparse subspace clustering (Elhamifar and Vidal 2009) involving sparse coding over a large size matrix needs 1 day on the same hardware platform. Our extensive experiments manifest that our strategy is both reliable and efficient.

3 Sparse Combinations Training

We describe our framework that learns sparse basis combinations. To extract usable data, we resize each frame into different scales as Cong et al. (2011) and uniformly partition each layer to a set of non-overlapping patches. All patches have the same size *in terms of pixel*. Corresponding regions in 5 continuous frames are stacked together to form a spatial-temporal cube. An example is illustrate in Fig. 2. This pyramid involves local information in fine-scale layers and more global structures in small-resolution ones.

With the spatial-temporal cubes, we compute 3D gradient features on each of them following (Kratz and Nishino 2009). These features in a video sequence are processed separately according to their spatial coordinates. Only features at the same spatial location in the video frames are used together for training and testing.

3.1 Learning Combinations on Training Data

For each cube location, 3D gradient features in all frames are denoted as $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \in \mathbb{R}^{p\times n}$, gathered temporally for training. Our goal is to find a basis combination set $\mathcal{S} = \{\mathbf{S}_1, \ldots, \mathbf{S}_K\}$ with each $\mathbf{S}_i \in \mathbb{R}^{p\times s}$ containing s dictionary basis vectors, forming a unique combination, where $s \ll q$. Here, the number of basis combinations K is adaptive to data, inferred by our algorithm. Each $\mathbf{S}_i$ belongs to a closed, convex and bounded set $\mathcal{C}$, which ensures column-wise unit norms to prevent over-fitting. Our sparse combination learning has to satisfy two requirements, namely effectiveness and efficiency representation.

Effective representation Our first requirement is to ensure a small reconstruction error for training data—each training sample in $\mathcal{X}$ should be constructed by at least one combination. It is coarsely expressed as

$$(\min_{\boldsymbol{\gamma}_j, \boldsymbol{\beta}_j} \sum_{i=1}^{K} \gamma_j^i \|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2) \le \lambda, \quad \forall j = 1, \ldots, n \tag{2}$$

where $\boldsymbol{\beta}_j = \{\boldsymbol{\beta}_j^1, \ldots, \boldsymbol{\beta}_j^K\}$ and $\boldsymbol{\gamma}_j = \{\gamma_j^1, \ldots, \gamma_j^K\}$. Each γ_j^i indicates whether or not the i^{th} combination $\mathbf{S}_i$ is chosen for data $\mathbf{x}_j$. $\boldsymbol{\beta}_j^i$ is the corresponding coefficient set for representing $\mathbf{x}_j$ with combination $\mathbf{S}_i$. The constraints $\sum_{i=1}^K \gamma_j^i = 1$ and $\gamma_j^i = \{0, 1\}$ require that only one combination is selected. The reconstruction error of any combination should be smaller than a threshold λ.

Efficient representation The other requirement is to make the total number of combinations K small to enable fast testing. Actually a surveillance video is inherently redundant for representation by a small number of combinations. The ideal minimal K can reflect how informative the data is.

Optimization framework We enable both effective and efficient representations in the unified model of

$$\begin{aligned}
&\min_{\mathcal{S}} K \\
&\text{s.t.} \min_{\boldsymbol{\gamma}_j, \boldsymbol{\beta}_j} \sum_{i=1}^{K} \gamma_j^i \|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2 \le \lambda, \ \forall j = 1, \ldots, n \\
&\sum_{i=1}^{K} \gamma_j^i = 1, \quad \gamma_j^i = \{0, 1\}, \quad \mathbf{S}_i \in \mathcal{C}, \ \forall i = 1, \ldots, K.
\end{aligned} \tag{3}$$

We seek a $\mathcal{S}$ whose combination number K is small and is with the reconstruction errors on training samples lower than λ.

3.2 Maximum Commonness Representation Strategy

We address the problem by the *maximum commonness representation strategy* (MCRS). It makes K naturally reduced by encouraging each combination to represent as many training samples as possible. We give a simple illustration in Fig. 3.

We solve Eq. (3) in iterations by maximizing the number of training samples represented by one combination each time. This process can quickly find dominating combinations in the first a few passes. Remaining training samples that cannot be well represented by current combinations are sent to the next round to gather residual maximum commonness.

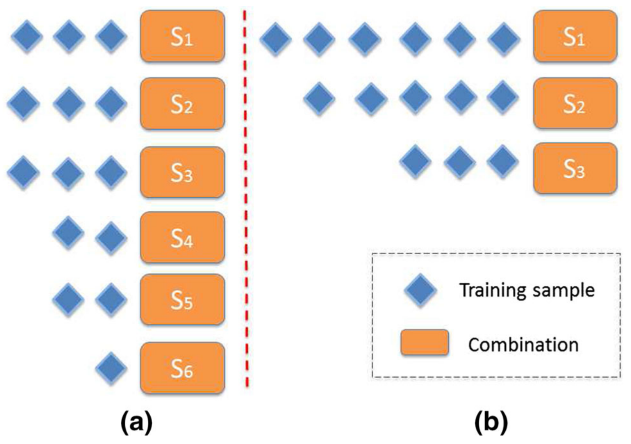

Fig. 3 An illustration of our MCRS. By maximizing the number of training samples that can be reconstructed by each combination, we naturally reduce the number K

This process ends until all training data are computed and bounded. The size of combinations K can reach a reasonably small value eventually.

To estimate a combination that represents as many training samples as possible, we propose a maximum commonness objective function as follows. We will prove it satisfies condition (2). Specifically, in the ith pass, given the leftover training data $\mathcal{X}_c(i) \subseteq \mathcal{X}$ that cannot be represented by previous combinations $\{\mathbf{S}_1, \ldots, \mathbf{S}_{i-1}\}$, we compute $\mathbf{S}_i$ to represent most data in $\mathcal{X}_c(i)$, expressed as

$$\max_{\boldsymbol{\gamma}} \sum_{j \in \Omega_c(i)} \gamma_j^i \tag{4}$$

and

$$\min_{\mathbf{S}_i, \boldsymbol{\gamma}, \boldsymbol{\beta}} \sum_{j \in \Omega_c(i)} \gamma_j^i (\|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2)$$
$$\text{s.t.} \quad \gamma_j^i = \{0, 1\}, \quad \mathbf{S}_i \in \mathcal{C}, \quad \forall i = 1, \ldots, K \tag{5}$$

where $\Omega_c(i)$ is the index set for $\mathcal{X}_c(i)$. Eq. (5) encourages $\mathbf{S}_i$ to represent training data. Eq. (4) maximizes commonness, which can be rewritten in the minimization form as

$$\min_{\boldsymbol{\gamma}} \sum_{j \in \Omega_c(i)} -\gamma_j^i. \tag{6}$$

Now, Eqs. (5) and (6) are combined to form a unified objective function, using a hyper-parameter λ, as

$$\min_{\mathbf{S}_i, \boldsymbol{\gamma}, \boldsymbol{\beta}} \sum_{j \in \Omega_c(i)} \gamma_j^i (\|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2) + \lambda (\sum_{j \in \Omega_c(i)} -\gamma_j^i)$$
$$\text{s.t.} \quad \gamma_j^i = \{0, 1\}, \quad \mathbf{S}_i \in \mathcal{C}, \quad \forall i = 1, \ldots, K \tag{7}$$

We name it as *maximum commonness model*. It is now easy to prove Eq. (7) satisfies condition (2). Specifically, if $\|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2 - \lambda \geq 0$, setting $\gamma_j^i = 0$ yields a smaller value compared to setting $\gamma_j^i = 1$. Contrarily, γ_j^i should be 1 if $\|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2 - \lambda < 0$, complying with condition (2).

Noted that $\gamma_j^i = 1$ indicates $\mathbf{x}_j$ is included in $\Omega_c(i)$ and $\gamma_j^q = 0$ where $q \leq j - 1$. Since $\mathbf{x}_j$ does not appear in next pass, we safely set $\gamma_j^q = 0$ where $q > j$. Therefore $\sum_{i=1}^K \gamma_j^i = 1$ holds after all passes.

3.2.1 Solver

There are a three variables in Eq. (7), namely $\mathbf{S}_i$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. We solve Eq. (7) by dividing it into two subproblems to iteratively update $\{\mathbf{S}_i, \boldsymbol{\beta}\}$ and $\boldsymbol{\gamma}$ using the following procedure.

Update $\{\boldsymbol{\beta}\}$ With fixed $\boldsymbol{\gamma}$ and $\mathbf{S}_i$, Eq. (7) becomes a quadratic function

$$L(\boldsymbol{\beta}, \mathbf{S}_i) = \sum_{j \in \Omega_c(i)} \gamma_j^i \|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2.$$
$$\text{s.t.} \quad \mathbf{S}_i \in \mathcal{C} \tag{8}$$

It can be solved by SVD decomposition with a global optimum guaranteed. But computation of SVD is heavy when the size of training samples is large. We resort to another iteration procedure to speed it up. We optimize $\boldsymbol{\beta}$ while fixing $\mathbf{S}_i$ for all $\gamma_j^i \neq 0$. These two steps alternate. The closed-form solution for $\boldsymbol{\beta}$ is

$$\boldsymbol{\beta}_j^i = (\mathbf{S}_i^T \mathbf{S}_i)^{-1} \mathbf{S}_i^T \mathbf{x}_j. \tag{9}$$

The resulting $\boldsymbol{\beta}_j^i$ can be considered as the orthogonal projection of $\mathbf{x}_j$ onto $\mathbf{S}_i$

Update $\{\mathbf{S}_i\}$ With fixed $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$, $\mathbf{S}_s^i$ finds its solution as

$$\mathbf{S}_i = \prod [\mathbf{S}_i - \delta_t \nabla_{\mathbf{S}_i} L(\boldsymbol{\beta}, \mathbf{S}_i)], \tag{10}$$

where δ_t is set to $1E-4$ and $\prod$ denotes projecting the basis to a unit column. Block-coordinate descent can converge to a global optimum due to its convexity (Bertsekas 1999). Therefore, the total energy for $L(\boldsymbol{\beta}, \mathbf{S}_i)$ decreases in iterations and finally converges.

Update $\boldsymbol{\gamma}$ With the $\{\mathbf{S}_i, \boldsymbol{\beta}\}$ output, for each $\mathbf{x}_j \in \mathcal{X}_c$, the objective function becomes

$$\min_{\gamma_j^i} \ \gamma_j^i \|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2 - \lambda \gamma_j^i$$
$$\text{s.t.} \ \ \gamma_j^i = \{0, 1\} \,. \tag{11}$$

The solution can be achieved based on the following Lemma.

Lemma 1 *In Eq. 11, γ_j^i has a closed-form solution*

$$\gamma_j^i = \begin{cases} 1 & if \ \ \|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2 \le \lambda \\ 0 & otherwise \end{cases} \tag{12}$$

Proof If $\|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2 - \lambda \ge 0$, setting $\gamma_j^i = 0$ yields a smaller value compared to setting $\gamma_j^i = 1$. Contrarily, γ_j^i should be 1 if $\|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2 - \lambda < 0$, □

Therefore, the update step guarantees condition (2).

3.2.2 Algorithm Summary and Analysis

Our maximum commonness representation strategy works well in practice. K is kept small by optimizing $\mathbf{S}$ using Eq. (7). The effectiveness condition (2) also strictly holds. This scheme reduces information overlap between combinations. The resulting energy of Eq. (7) does not increase in iterations, since all subproblems find global optima.

Our MCRS framework is described in Algorithm 1. The initial dictionary $\mathbf{S}_i$ in each pass is calculated by clustering training data $\mathcal{X}_c$ via K-means with s centers.

Our algorithm is controlled by λ—the upper bound of reconstruction errors. Reducing it could lead to a larger K. Our approach is expressive because all training normal event patterns are represented with controllable reconstruction errors under condition (2).

Algorithm 1 Maximum Commonness Representation Strategy (MCRS)

```
Input: $\mathcal{X}$, current training features $\mathcal{X}_c = \mathcal{X}$
initialize $\mathcal{S} = \emptyset$ and $i = 1$
repeat
    repeat
        Optimize $\{\boldsymbol{\beta}\}$ and $\{\mathbf{S}_i\}$ with Eqs. (9) and (10)
        Optimize $\{\boldsymbol{\gamma}\}$ using Eq. (12)
    until Eq. (8) converges
    Add $\mathbf{S}_i$ to set $\mathcal{S}$
    Remove computed features $\mathbf{x}_j$ with $\gamma_j^i = 1$ from $\mathcal{X}_c$
    $i = i + 1$
until $\mathcal{X}_c = \emptyset$
Output: $\mathcal{S}$
```

4 Birth-and-Death Combination Online Learning

The sparse combination learning step in Sect. 3 requires to keep all data in memory. Its computational cost is polynomial with respect to data scale. To deal with large data and dynamic scene, we further improve the online system with constant memory consumption and linear computation complexity with respect to the number of training data.

The dynamic scene problem can be explained this way—a normal pattern can become abnormal over time. For example, in a building entrance, one person moving inside is normal in the morning, but it can be abnormal after people leave in the afternoon.

In traditional online dictionary learning (Szabo et al. 2011; Mairal et al. 2010; Zhao et al. 2011), the dictionary is updated by assigning recent samples with large weights to capture normal-pattern evolution. However, our representation is completely different under the combination definition. We introduce the principle of combination birth-and-death where new combinations are formed to capture new normal patterns, while old combinations could be dead when describing outdated patterns.

Our new strategy is to separate training data $\mathcal{X}$ into many mini-batches. Each mini-batch contains h samples. $\{\mathbf{x}_{(k-1)h+1}, \ldots, \mathbf{x}_{kh}\}$ is contained in the kth mini-batch $\mathcal{X}_k$. We denote the learned sparse combinations after processing the kth mini-batch as $\mathcal{T}_k$. Obviously, $\mathcal{T}_1$ can be obtained by sparse combination learning in Algorithm 1. Then we process $\mathcal{X}$ in an online manner for the following mini-batch $\mathcal{X}_{k+1}$.

4.1 Optimization for Online Training

We denote $\mathcal{T}_k = \{\mathbf{S}_1, \ldots, \mathbf{S}_{v_k}\}$, where v_k is the number of combinations for $\mathcal{T}_k$. Each combination has an active score to indicate its current activeness level. For dynamic scenes, we remove these outdated combinations according to the active score. The ith active score is denoted as φ_i.

When next mini-batch $\mathcal{X}_{k+1} = \{\mathbf{x}_{kh+1}, \ldots, \mathbf{x}_{(k+1)h}\}$ arrives, we first update $\mathcal{T}_k$. Then we create new combinations if there exist samples from $\mathcal{X}_{k+1}$ that cannot be represented by $\mathcal{T}_k$. We remove some combination if their active scores are less than a threshold ϵ.

4.1.1 Updating Existing Combinations in $\mathcal{T}_k$

We update $\mathcal{T}_k$ by first evaluating whether the new samples can be represented by $\mathcal{T}_k$ or not. For data $\mathbf{x}_j \in \mathcal{X}_{k+1}$, if there exists $\mathbf{S}_i \in \mathcal{T}_k$, which satisfies $\min_{\boldsymbol{\beta}_j^i} \|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2 < \lambda$, we assign $\gamma_j^i = 1$. Otherwise, we set it as zero. Then $\mathcal{T}_k$ is updated by minimizing the function for each $\mathbf{S}_i$ of

$$W(\mathbf{S}_i) = \sum_{j=1}^{(k+1)h} \gamma_j^i \|\mathbf{S}_i \boldsymbol{\beta}_j^i - \mathbf{x}_j\|_2^2 \quad \text{s.t.} \quad \mathbf{S}_i \in \mathcal{C} \tag{13}$$

Similar to Eqs. (8), (13) can be solved via block-coordinate descent update as

$$\mathbf{S}_i = \prod[\mathbf{S}_i - \delta_t \nabla_{\mathbf{S}_i} W(\mathbf{S}_i)], \tag{14}$$

where δ_t is set to $1E-4$ and $\prod$ denotes projecting the basis to a unit column. According to Bertsekas (1999), by solving Eq. (14) iteratively, we can reach a global optimum for Eq. (13) due to its convexity. The derivative part in Eq. (14) is

$$\nabla_{\mathbf{S}_i} W(\mathbf{S}_i) = \mathbf{S}_i M_j^{k+1} - E_j^{k+1}, \tag{15}$$

where

$$M_i^{k+1} = \sum_{j=1}^{(k+1)h} \gamma_j^i \boldsymbol{\beta}_j^i (\boldsymbol{\beta}_j^i)^T,$$
$$E_i^{k+1} = \sum_{j=1}^{(k+1)h} \gamma_j^i \mathbf{x}_j (\boldsymbol{\beta}_j^i)^T. \tag{16}$$

Following Mairal et al. (2010) and Lu et al. (2013b), we record M_i and E_i as two auxiliary matrices in each mini-batch process. M_i^{k+1} and E_i^{k+1} are updated with simple computation as

$$M_i^{k+1} = M_i^k + \sum_{j=kh+1}^{(k+1)h} \gamma_j^i \boldsymbol{\beta}_j^i (\boldsymbol{\beta}_j^i)^T,$$
$$E_i^{k+1} = E_i^k + \sum_{j=kh+1}^{(k+1)h} \gamma_j^i \mathbf{x}_j (\boldsymbol{\beta}_j^i)^T. \tag{17}$$

With that two auxiliary matrices, we do not need to store all information for history data. For large data, it only use constant running time and memory for combination update.

4.1.2 Generating New Combinations

We remove those training samples in $\mathcal{X}_{k+1}$ that can be represented by existing combinations in $\mathcal{T}_k$ described in previous section, and form a new data set $\overline{\mathcal{X}}_{k+1}$. It is a subset whose samples cannot be represented by $\mathcal{T}_k$. Then we apply our standard sparse combination leaning solver in Algorithm 1 into create new combinations.

4.1.3 Removing Outdated Combinations

For each combination, we compute an active score to measure its activeness. We define the active score of the i^{th} combination as

$$\varphi_i^t = \frac{\sum_{j=1}^t \rho^j \gamma_j^i}{\rho^t \sum_{j=1}^t \gamma_j^i}, \tag{18}$$

where ρ is a factor and ρ^j is the power j of ρ. If $\rho > 1$, samples existing for long time result in small weights. If a combination whose most samples are old, its active score would be small. When $\rho = 1$, the active score is 1 for consistency. We also define an auxiliary normalized factor $C^i(t)$ as

$$C^i(t) = \sum_{j=1}^t \gamma_j^i = C^i(t-1) + \gamma_t^i. \tag{19}$$

Considering Eqs. (18) and (19), we have the update function

$$\varphi_i^t = \frac{C^i(t-1)}{\rho C^i(t)} \varphi_i^{t-1} + \frac{\gamma_t^i}{C^i(t)} \tag{20}$$

Algorithm Our framework works for both dynamic scenes ($\rho > 1$) and stationary patterns ($\rho = 1$). The entire birth-and-death online learning procedure is listed in Algorithm 2.

Algorithm 2 Online birth-and-death Combination Learning

```
Input: $\mathcal{T}_k = \{\mathbf{S}_1, \ldots, \mathbf{S}_{v_k}\}$, mini-batch data $\mathcal{X}_{k+1} = \{\mathbf{x}_{kh+1}, \ldots, \mathbf{x}_{(k+1)h}\}$, history statistics $M_i^k$, active score $\{\varphi_1^{kh}, \ldots, \varphi_{v_k}^{kh}\}$, active normalized factor $\{C^1(kh), \ldots, C^{v_k}(kh)\}$
for $j = kh+1 \to (k+1)h$ do
    for $i = 1 \to v_k$ do
        if $\min_{\boldsymbol{\beta}_j^i} \|\mathbf{x}_j - \mathbf{S}_i \boldsymbol{\beta}_j^i\|_2^2 < \lambda$ then
            $\gamma_j^i = 1$; break.
        end if
    end for
end for
for $i = 1 \to v_k$ do
    Update $M_i^{k+1}$, and $E_i^{k+1}$ using Eq. (17).
    Update $S_i$ using Eq. (13).
end for
Set $\overline{\mathcal{X}}_{k+1} = \mathcal{X}_{k+1}$.
for $j = kh+1 \to (k+1)h$ do
    if $\sum_i^{v_k} \gamma_j^i = 1$ then
        Remove $\mathbf{x}_j$ from $\overline{\mathcal{X}}_{k+1}$.
    end if
end for
Generate new combinations via Alg. 1 for $\overline{\mathcal{X}}_{k+1}$.
Generate $M_i^{k+1}$ and $E_i^{k+1}$, $i = v_k+1, \ldots \widehat{v}_{k+1}$.
for $j = kh+1 \to (k+1)h$ do
    update $\{C^j(1), \ldots, C^j(\varphi_{v_k})$ using Eq. (19)
    update $\{\varphi_1^j, \ldots, \varphi_{v_k}\}^j$ using Eq. (20)
end for
for $i = 1 \to \widehat{v}_{k+1}$ do
    if $\varphi_i^{(k+1)h} < \epsilon$ then
        remove $i^{th}$ Combination
    end if
end for
Output: $\mathcal{T}_{k+1} = \{\mathbf{S}_1, \ldots, \mathbf{S}_{v_{k+1}}\}$.
```

4.2 Analysis for Online Training

In our online process, it is not necessary to keep all data in memory. Instead, we only record history data statistics M and E, which result in constant memory consumption. In terms of running time, the computation complexity of our online approach is $O(UN/h)$, where U is the upper bound of min-batch processing time. It means the computation complexity is linear with respect to data scale N in our online algorithm, compared with the polynomial form for the batch algorithm. In the following experiment section, we demonstrate that the online solver can reduce training cost significantly for a large amount of training data with only minor accuracy compromise.

5 Testing

Given the learned sparse combinations $\mathcal{S} = \{\mathbf{S}_1 \dots \mathbf{S}_K\}$, in the testing phase with new data $\mathbf{x}$, we check if there exists a combination in $\mathcal{S}$ fitting the reconstruction error upper bound. It can be quickly achieved by checking the least square error for each $\mathbf{S}_i$:

$$\min_{\boldsymbol{\beta}^i} \|\mathbf{x} - \mathbf{S}_i \boldsymbol{\beta}^i\|_2^2 \quad \forall\, i = 1, \dots, K \tag{21}$$

It is a standard quadratic function with the optimal solution

$$\widehat{\boldsymbol{\beta}}^i = (\mathbf{S}_i^T \mathbf{S}_i)^{-1} \mathbf{S}_i^T \mathbf{x}. \tag{22}$$

The reconstruction error in $\mathbf{S}_i$ is

$$\|\mathbf{x} - \mathbf{S}_i \widehat{\boldsymbol{\beta}}^i\|_2^2 = \|(\mathbf{S}_i(\mathbf{S}_i^T \mathbf{S}_i)^{-1} \mathbf{S}_i^T - \mathbf{I}_p)\mathbf{x}\|_2^2, \tag{23}$$

where $\mathbf{I}_p$ is a $p \times p$ identity matrix. To further simplify computation, we define an auxiliary matrix $\mathbf{R}_i$ for each $\mathbf{S}_i$:

$$\mathbf{R}_i = \mathbf{S}_i(\mathbf{S}_i^T \mathbf{S}_i)^{-1} \mathbf{S}_i^T - \mathbf{I}_p. \tag{24}$$

Reconstruction error for $\mathbf{S}_i$ is accordingly $\|\mathbf{R}_i \mathbf{x}\|_2^2$. If it is small, $\mathbf{x}$ is regarded as a normal event pattern. The final testing scheme is summarized in Algorithm 3.

Algorithm 3 Testing with Sparse Combinations

```
Input: $\mathbf{x}$, auxiliary matrices $\{\mathbf{R}_1, \dots, \mathbf{R}_K\}$ and threshold $T$.
for $j = 1 \rightarrow K$ do
    if $\|\mathbf{R}_k \mathbf{x}\|_2^2 < T$ then
        return normal event.
    end if
end for
return abnormal event.
```

It is noted that the first a few dominating combinations represent the largest number of normal event features, which enable us to determine positive data quickly. In our experiments, the average combination checking ratio is 0.325, which is the number of combinations checked divided by the total number K. Also, our method can be easily accelerated via parallel processing to achieve $O(1)$ complexity although it is already very fast.

5.1 Comparison with ℓ_1 norm Minimization

ℓ^1-norm approximation is another solution to speed up Eq. (1). However, it still assumes that a good basis combination from dictionary atoms can be found. This model does not consider the fact that in surveillance videos, normal patterns are highly redundant in terms of the level of information where a small number of combinations can already represent them. Differently, our sparse combination learning encourages that the number of basis combinations is adaptive to the complexity of data. Our testing only needs least square (by matrix multiplication) operations. We will compare our method with ℓ^1-norm minimization solvers in terms of efficiency in the following section.

6 Experiments

In this section, we empirically demonstrate that our model is suitable to represent general surveillance videos. We apply our method to different datasets. Quantitative comparisons are reported.

6.1 System Setting

The size of $\mathbf{S}_i \in \mathbb{R}^{p \times s}$ controls the sparsity level. We experimentally set $s = 0.1 \times p$ where p is the data dimension. λ in Eq. (2) is the error upper bound, set to 0.04 in experiments.

Given the input video, we resize each frame to 3 scales with 20×20, 30×40, and 120×160 pixels respectively and uniformly partition each layer to a set of non-overlapping 10×10 patches, leading to 208 sub-regions for each frame in total shown in Fig. 2. Corresponding sub-regions in 5 continuous frames are stacked together to form a spatial-temporal cube, each with resolution $10 \times 10 \times 5$. We compute 3D gradient features on each of them following (Kratz and Nishino 2009). Those gradients are concatenated to a 1500-dimension feature vector for each cube and are then reduced to 100 dimensions via PCA. Normalization is performed to make them mean 0 and variance 1.

In frame-level detection, we compute an abnormal indicator V by summing the number of cubes in each scale with weights. It is defined as $V = \sum_{i=1}^{z} 2^{z-i} v_i$, where v_i is the number of abnormal cubes in scale i. The top scale is with

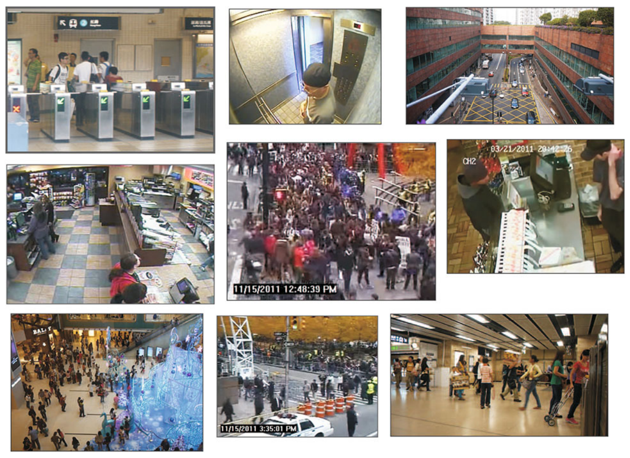
Fig. 4 A few frames from the surveillance videos used for verification

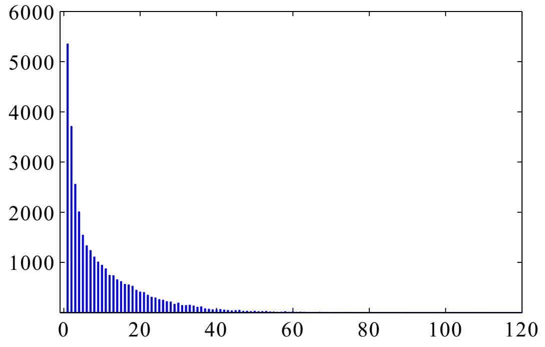

Fig. 5 Different numbers of basis combinations to represent normal events in 31,200 groups (x-axis: K; y-axis: number of groups that use K combinations)

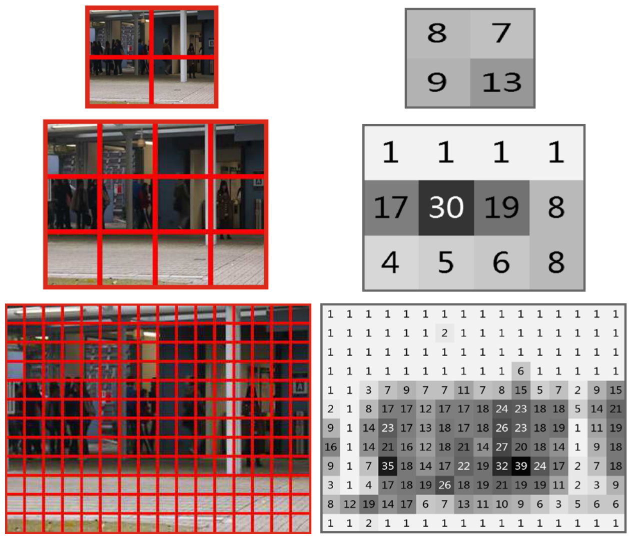

Fig. 6 Spatial distribution of combination numbers to represent normal structures in the Avenue data

index 1 while the bottom one is with z ($z = 3$ in our experiment). For pixel or region level detection, abnormal pixels are the ones whose corresponding cubes (in any scale) are abnormal. In the online solver, we fix the size of mini-batch to $h = 200$. All experiments are conducted using MATLAB. In what follows, "Our (online)" and "Our (batch)" in reporting results refer to those achieved by online solver (Sect. 4) and batch solver (Sect. 3) respectively.

6.2 Verification of Sparse Combinations

Surveillance videos consist of many redundant patterns. For example, in subway exit, people generally move in similar directions. These patterns share information coded in our sparse combinations. To verify it, we collect 150 normal event surveillance videos with a total length of 107.8 h. The videos are obtained from sources including dataset UCSD Ped1 (Mehran et al. 2009), Subway datasets (Adam et al. 2008) (excluding abnormal event frames), 68 videos from YouTube, and 79 videos we captured. The scene includes subway, mall, traffic, indoor, elevator, square, etc. We show a few example frames in Fig. 4.

Each video contains 208 regions as illustrated in Fig. 6. With the 150 different videos, we gather a total of 31,200 (208×150) groups of cube features with each group corresponding to a set of patches (cubes). They are used separately to verify the combination model. Each group contains 6,000-120,000 features. The number of combinations for each group is denoted as K. We show in Fig. 5 the distribution of K in the 31,200 groups. Its mean is 9.75 and variance is 10.62, indicating 10 combinations are generally enough in our model to represent events. The largest K is 108. About 99% of the Ks are smaller than 45.

We illustrate the K distributions spatially in the Avenue data (described below) in Fig. 6. Many regions only need 1 combination because they are static. Largely varying patches may need dozens of combinations to summarize normal events. The statistical regression error is as small as $0.0132 \pm 1.38E{-}4$, which indicates our dictionaries contain almost all normal patterns (Fig. 7).

6.3 Avenue Data Benchmark

We construct a new avenue dataset for evaluation. In comparison to our conference version, we extend the avenue dataset from 23 videos to 37 videos. The videos are captured in a campus with 30652 (15328 for training and 15324 for testing) frames in total. Frame resolution is 360×640. The training videos capture normal situations. Testing videos include both normal and abnormal events. Three abnormal samples are shown in Fig. 8.

Our dataset includes a few realistic scenarios that are not contained in pervious datasets. First, some clips are with slight camera shake (frames 1051–1100 in testing video 2).

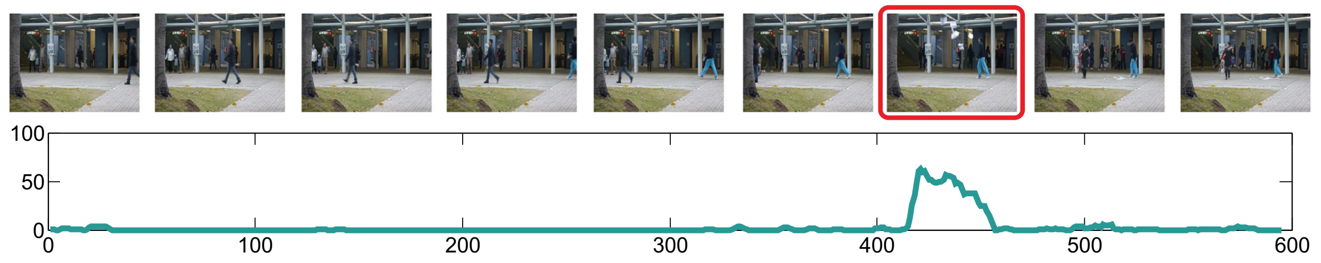

Fig. 7 Detection results in a video sequence. The bottom plot is the response. A peak appears when an abnormal event-paper throwing-happens. The x value indexes frames and y-index denotes response strength

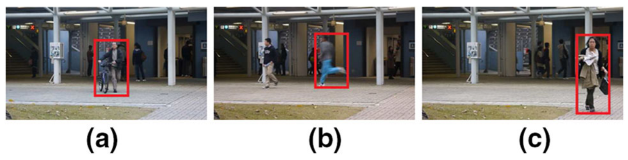

Fig. 8 Representative abnormal events in Avenue Dataset. **a** Abnormal object, **b** strange action, **c** wrong direction

Second, a few outliers are involved in training data, which require the training process robust enough. These are common problems in real world videos.

Besides frame-level evaluation, we further evaluate abnormal events in the region level. We label spatial locations of abnormal events using rectangular regions similar to the bounding boxes in VOC Pascal labeled data, and define abnormality in a frame as

$$\frac{A_d \cap A_g}{A_d \cup A_g} \geq \upsilon. \tag{25}$$

where A_d and A_g are the detected region and ground truth abnormal event region respectively and υ is a threshold. For the frames containing no abnormal event, we have $A_d \cap A_g = 0$. In this case, we specially define the result value as 1. The accuracy is the percentage of frames satisfying Eq. (25). We report our accuracy values under thresholds in {0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8} in Table 1.

A video sequence and its abnormal event detection result are demonstrated in Fig. 7. Figure 9 contains three important frames and their abnormal event regions in two image scales.

This dataset is with stationary patterns. We set $\rho = 1$ for the active score in Eq. (18) with $\epsilon = 0.01$. We list the detection statistics in Table 1. The performance of our methods is satisfactory with the average detection speed as quick as **964.8** frames per second.

6.4 Subway Dataset

We conduct quantitative comparison with previous methods on the Subway dataset (Adam et al. 2008). The videos are 2 h long in total, containing 209,150 frames with size 512 × 384. There are two types of videos, i.e., "exit gate" and "entrance gate" videos. We also set $\rho = 1$ in the active score and $\epsilon = 0.01$ in Eq. (18).

6.4.1 Exit Gate

The subway exit surveillance video contains 19 different types of unusual events, such as walking in the wrong direction and loitering near the exit. The video sequence in the first 15 min is used for training. This configuration is the same as those in Kim and Grauman (2009) and Zhao et al. (2011).

The abnormal event detection results for a few frames are shown in Fig. 10. Table 2 lists the comparison with other methods. Our false alarm rate is low mainly because each combination can construct many normal event features, thus reducing the chance of constructing an abnormal structure with a small error. This representation tightens feature modeling and makes it not that easy to misclassify abnormality

Table 1 Detection accuracy (in %) under different threshold υ in the Avenue dataset

υ	0.2	0.3	0.4	0.5	0.6	0.7	0.8	Mean
Sparse combination (Batch solver)	70.0	67.3	63.3	59.3	57.5	55.7	54.4	61.0
Sparse combination (Online solver)	68.3	65.2	62.2	56.1	54.2	52.5	51.2	58.5
Subspace clustering (K = 10)	58.7	52.5	47.1	45.5	42.9	39.1	36.4	46.0
Subspace clustering (K = 50)	50.6	47.4	43.5	41.0	38.7	36.8	34.1	41.7
Subspace clustering (best K = 13)	60.2	54.4	49.7	47.3	44.8	40.5	38.3	47.9

Subspace clustering (K = 10), (K = 50) and (best K = 13) means using subspace number 10, 50 and 13 respectively, where "K = 13" has best performance

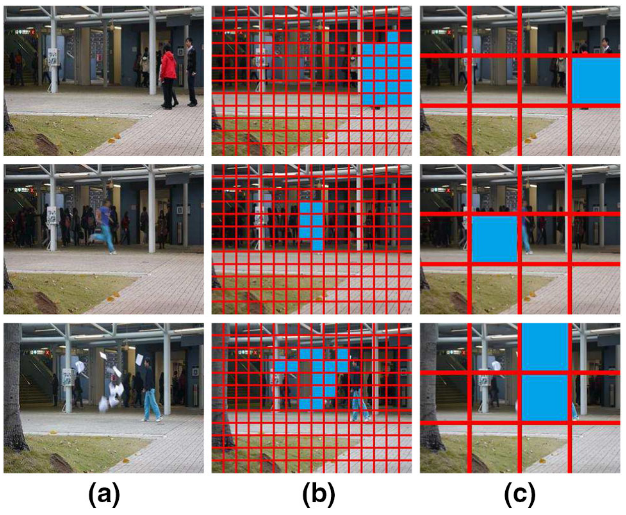
Fig. 9 Three abnormal events and their corresponding abnormal patches under two different scales in the Avenue dataset. **a** Events, **b** maps, **c** maps

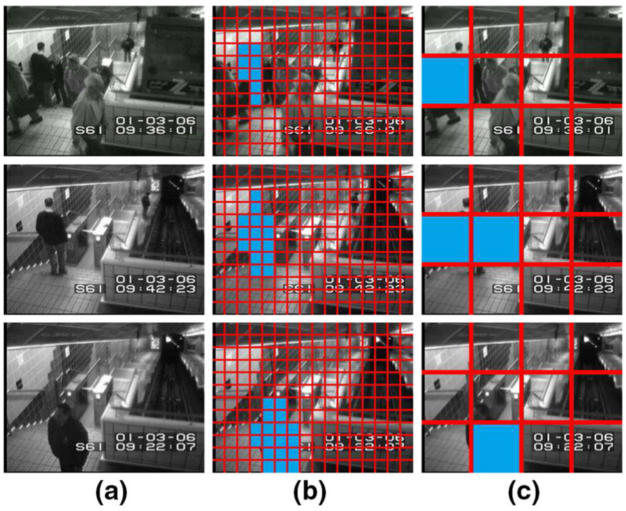

Fig. 10 Subway dataset (exit-gate): three abnormal events and their corresponding detection maps in two different scales. **a** Events, **b** maps, **c** maps

as normal events. In this dataset, our combination number K varies from 1 to 56 for different cube features.

6.4.2 Entrance Gate

In this video, again, the first 15 min are used for training. Detection statistics are listed in Table 3. Our results are comparable to those of Zhao et al. (2011), Kim and Grauman (2009) and Cong et al. (2011). The proposed method yields high detection rates together with low false alarm.

Table 2 Comparison with other sparsity-based methods (Zhao et al. 2011; Cong et al. 2011) on the Exit-Gate subway dataset

	WD	LT	MISC	Total	FA
Ground truth	9	3	7	19	0
Zhao et al. (2011)	9	3	7	19	2
Kim and Grauman (2009)	9	3	7	19	3
Cong et al. (2011)	9	–	–	–	2
Subspace (K = 10)	6	3	5	14	4
Subspace (K = 50)	5	2	5	12	5
Subspace (best, K = 15)	7	3	5	15	4
Ours (online)	9	3	7	19	3
Ours (batch)	9	3	7	19	2

WD wrong direction; *LT* loitering; *FA* false alarm; "-": results not provided. "Subspace" represents replacing our combination learning by subspace clustering (Elhamifar and Vidal 2009). Subspace (K = 10), (K = 50) and (best K = 15) means using subspace number 10, 50 and 13 respectively, where "K = 15" has best performance

6.4.3 Running Time Comparison

We compare our system with other sparse dictionary learning based methods (Zhao et al. 2011; Cong et al. 2011) in terms of running time on the Subway dataset in Table 4. The speed of methods (Zhao et al. 2011; Cong et al. 2011) is reported in their respective papers. The discrepancy is huge.

6.5 UCSD Ped1 Dataset

The UCSD Ped1 dataset (Mahadevan et al. 2010) provides 34 short clips for training, and another 36 clips for testing. All testing clips have frame-level ground truth labels, and 10 clips have pixel-level ground truth labels. There are 200 frames in each clip.

Our configuration is similar to that of Mahadevan et al. (2010). That is, the performance is evaluated on frame- and pixel-levels. We show the results via ROC curves, Equal Error Rate (EER), and Equal Detected Rate (EDR).

6.5.1 ROC Curve Comparison

According to Mahadevan et al. (2010) in frame-level detection, if a frame contains at least one abnormal pixel, it is considered as successful detection. In our experiment, if a frame contains one or more abnormal patches, we label it as an abnormal event. For frame-level evaluation, we alter frame abnormality threshold to produce a ROC curve shown in Fig. 11. Our method has a reasonably high detection rate when the false positive value is low. It is vital for practical detection system development.

In pixel level evaluation, a pixel is labeled as abnormal, if and only if the regions it belongs to in all scales are abnormal. We alter threshold for all pixels. Following Mahadevan

Table 3 Comparison using the subway-entrance video with several previous methods

	WD	NP	LT	II	MISC	Total	FA
GT	26	13	14	4	9	66	0
Zhao et al. (2011)	25	**9**	**14**	4	8	**60**	5
Kim and Grauman (2009)	24	8	13	4	8	57	6
Cong et al. (2011)	21	6	–	–	–	–	4
Subspace (K = 10)	21	6	9	3	7	46	7
Subspace (K = 50)	20	6	8	3	6	43	8
Subspace (best K = 13)	21	6	9	4	7	47	6
Ours (online)	24	7	12	4	7	54	5
Ours (batch)	**25**	7	13	**4**	**8**	57	**4**

GT ground truth; *WD* wrong direction; *NP* no payment; *LT* loitering; *II* irregular interactions; *MISC* misc; *FA* false alarm (desire to be small). "-" means results are not provided. Subspace: replacing our combination learning by subspace clustering (Elhamifar and Vidal 2009). Subspace (K = 10), (K = 50) and (best K = 13) means using subspace number 10, 50 and 13 respectively, where "K = 13" has best performance

Table 4 Testing time comparison on the Subway dataset

	SPF	Platform	CPU	Memory
Zhao et al. (2011)	2	MATLAB 7.0	2.6 GHz	2.0 GB
Cong et al. (2011)	4.6	–	2.6 GHz	2.0 GB
Ours (online)	**0.00102**	MATLAB 2012	3.4 GHz	8.0 GB
Ours (batch)	**0.00098**	MATLAB 2012	3.4 GHz	8.0 GB

"SPF" stands for seconds per frame

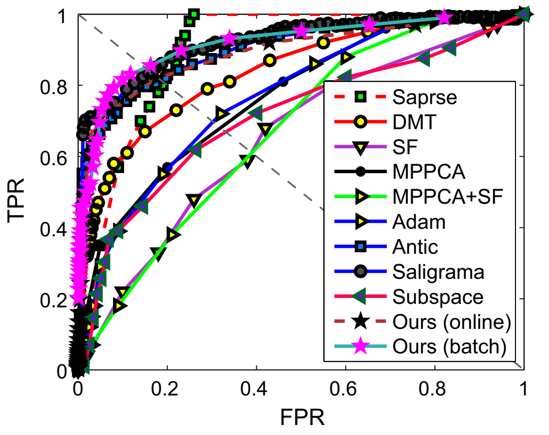

Fig. 11 Frame-level comparison of the ROC curves on UCSD Ped1 dataset. Method abbreviation: MPPCA+SF (Mahadevan et al. 2010), SF (Mahadevan et al. 2010), MDT (Mahadevan et al. 2010), Sparse (Cong et al. 2011), Saligrama (Saligrama and Chen 2012a), Antic (Antic and Ommer 2011), Adam (Adam et al. 2008) Subspace: replacing our combination learning by subspace clustering (Elhamifar and Vidal 2009)

et al. (2010), if more than 40% of truly anomalous pixels are detected, the corresponding frame is considered as being correctly detected. We show the ROC curve in Fig. 12. Besides all methods that are compared in Mahadevan et al. (2010), we also include the performance of subspace clustering (Elhamifar and Vidal 2009). Our method achieves satisfactory performance.

We demonstrate the influence of parameters s and λ in Tables 5 and 6 . Reasonable ranges of λ and s are 0.01–0.05 and $0.05p$ $0.2p$ (p is the feature dimension) respectively.

6.5.2 EER and EDR

Different parameters could affect detection rates. Following Mahadevan et al. (2010), we obtain these rates when the false positive number equals to the missing value. They are called equal error rate (EER) and equal detected rate (EDR). We compute the area under the ROC curve (AUC). The running time is shown in Table 7. The detection time per frame and the configuration of Mahadevan et al. (2010), Cong et al. (2011) and Antic and Ommer (2011) are obtained from the original papers. We report EER, ERD and AUC in the pixel-level comparison (Table 8) and calculate EER and AUC in the frame-level (Table 9). Our results are also with high quality regarding these measures (Fig. 13).

6.5.3 Pixel-Wise Performance Improvement

Unlike image parsing (Antic and Ommer 2011) that employs pixel-wise processing, our approach works in the patch level. As shown in Fig. 14b, the regional result is sufficient to capture abnormal events. To finally improve our performance under the pixel-level metric, we apply a simple foreground bilateral filtering on the regional abnormality map. It is to

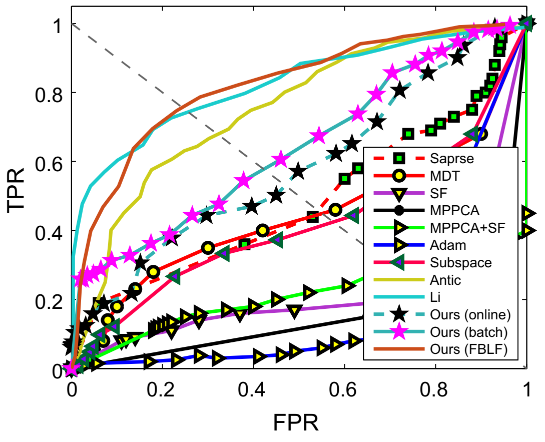

Fig. 12 Pixel-level comparison of the ROC curves in UCSD Ped1 dataset. Method abbreviation: MPPCA+SF (Mahadevan et al. 2010), SF (Mahadevan et al. 2010), MDT (Mahadevan et al. 2010), Sparse (Cong et al. 2011), Saligrama (Saligrama and Chen 2012a), Antic (Antic and Ommer 2011), Li (Li et al. 2014), Adam (Adam et al. 2008), Subspace: replacing our combinations learning by subspace clustering (Elhamifar and Vidal 2009)

Table 5 ACU performance on Ped1 dataset (frame-level) with respect to different s

s	$0.01p$	$0.05p$	$0.1p$	$0.2p$	$0.5p$	$0.8p$
AUC	39.6%	84.7%	91.8%	87.1%	56.3%	35.5%

p is the feature dimension. We fix $\lambda = 0.04$

Table 6 ACU performance on Ped1 dataset (frame-level) with respect to different λ, we fix $s = 0.1 \times p$, where p is the feature dimension

λ	0.001	0.005	0.01	0.05	0.1	0.5
AUC	45.1%	74.1%	88.3%	92.0%	87.5%	30.7%

joint filter the regional map (Fig. 14b) with the foreground map (c). The result is shown in (d), produced almost instantly.

We show our decent pixel-level result in the last column of Table 8. We estimate the background map by simple frame average. The foreground bilateral filtering takes 0.007 second per frame in average, achieving 130–150 FPS.

6.6 UCSD Ped2 Data (Mahadevan et al. 2010)

UCSD Ped2 dataset was proposed in Mahadevan et al. (2010). The dataset contains 16 training video clips and 12 testing clips. The frame resolution is 360 × 240. We implement our method following previous setting. Quantities of the area under ROC curves (AUCs) are reported in Table 10. Our results are reasonable.

We also compare the results in the pixel level. We list AUC of pixel-level detection in Table 11. Our method outperforms others except for that of Antic and Ommer (2011). Our method can process 978.45 frames per second in comparison to 0.2–0.1 frames per second using the method of Antic and Ommer (2011).

6.7 UMN Data [1]

There are three scenes in the UMN dataset [1], which contains crowds. People running suddenly produces an abnormal event. Our goal is to detect these abnormal frames. There are 7739 frames with resolution 240 × 320. Following Saligrama and Chen (2012b), we use the first 600 normal frames of each scene as training data and others for testing. Our method can also achieve state-of-the-art results, as shown in Table 12. A few representative frames are shown in Fig. 13. We achieve 981.75 FPS detection speed. UMN dataset is a small scale dataset. So we do not use our online solver.

6.8 Long Video: NYPD Dataset

We carry out experiment on a long video sequence to test the efficiency of our online solver. The video we use is a 7.5-h street surveillance one[1] with busy crowd and vehicles. A new abnormal event dataset is built on this video. We name it NYPD Dataset. Unlike subway dataset with many static scenes, 96.6% of the frames in the NYPD dataset are dynamic and moving ones. In addition, the illumination and environment-lighting change drastically from 4:32pm to 12:00am (from daytime to night) (Fig. 15).

This dataset slightly changes the normal patterns over time due to the crowd. In online procedure, we set $\rho = e^{\log(2)/\varsigma}$ to compute the active score in Eq. (18), where ς is the frame number in 10 min. It sets the weight of current samples 2 times larger than the one 10 min ago. In this dataset, we have $\rho = 1.000046210879727$. We set the threshold as $\epsilon = 0.001$. Thus normal patterns of 2 h ago are removed. We also experimented with $\rho = 1$ that assumes stationary patterns.

The dataset includes 853,200 frames with resolution 240 × 320. We select a 7-h normal video to form a training dataset – representative frames are demonstrated in Fig. 16. The testing data contain 20 min clip including 31 abnormal events—examples are shown in Fig. 17. In the testing data, abnormal and normal events take 6 and 14 min respectively. We evaluate the performance in region-level where bounding boxes of abnormal events are manually labeled.

The evaluation scheme is the same as that for the Avenue dataset. The accuracy comparison between our online and batch solvers is given in Table 13. We also compare our results with sparse subspace clustering. Unfortunately, the standard process is slow. Using the publicly available code does not output result after 70 days on a 24-core server. Therefore, we resort to a faster version of scalable sparse

[1] http://www.youtube.com/watch?v=sTvpzN4CldE

Table 7 Testing time comparison on the UCSD Ped1 dataset

	SPF	Platform	CPU	Memory
Mahadevan et al. (2010)	25	–	3.0 GHz	2.0 GB
Cong et al. (2011)	3.8	–	2.6 GHz	2.0GB
Antic and Ommer (2011)	5–10	MATLAB	–	–
Ours (online)	**0.00098**	MATLAB	3.4 GHz	8.0 GB
Ours (batch)	**0.00096**	MATLAB	3.4 GHz	8.0 GB

SPF stands for seconds per frame

Table 8 Comparison of pixel-level EDR and AUC curves on the UCSD Ped1 dataset

	Adam et al. (2008)	Antic and Ommer (2011)	Elhamifar and Vidal (2009)	Kaltsa et al. (2015)	Li et al. (2014)	Cheng et al. (2015)	Xu et al. (2015)	Ours (online)	Ours (batch)	Ours + Xu et al. (2015)	Ours (FBLF)	Ours + Xu et al. (2015) (FBLF)
EDR (%)	24	68	39.3	79	74.3	63.4	62.1	54.5	59.1	66.5	75.7	76.5
AUC (%)	13.3	76	43.2	80	82.7	72.2	67.2	58.8	63.8	75.5	82.1	84.1

"FBLF" means our abnormality maps are processed with foreground bilateral filter. "Ours + Xu et al. (2015)" means replacing 3D gradient features by autoencoding feature proposed in Xu et al. (2015)

Table 9 Comparison of frame-level EER and AUC curves on the UCSD Ped1 dataset

	MDT (Mahadevan et al. 2010)	Sparse (Cong et al. 2011)	Saligrama (Saligrama and Chen (2012a)	Antic (Antic and Ommer 2011)	Xu et al. (2015)	SC (K = 10)	SC (K = 50)	SC (K = 14)	Ours (online)	Ours (batch)	Ours + Xu et al. (2015)
EER (%)	25	19	16	18	16	29.6	30.2	27.4	19	15	14
AUC (%)	81.8	86	92.7	91	92.1	68.4	67.9	70.2	89.1	91.8	93.8

"Ours + Xu et al. (2015)" means replacing 3D gradient features by autoencoding feature proposed in Xu et al. (2015). SC (K = 10), (K = 50) and (K = 14) means subspace clustering method (Elhamifar and Vidal 2009) using subspace number 10, 50 and 14 respectively, where "K = 14" has best performance

subspace clustering (Peng et al. 2013). We report our accuracy under overlap thresholds $\{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$ in Table 13. It demonstrates that our birth-and-death online solver helps handle the dynamic scene problem. The testing phase using our sparse combination learning reaches 1073.8 FPS.

6.9 Training Cost

To demonstrate the training efficiency of our online solver, the training cost is reported and discussed. Training time with respect to different data scales on each dataset is plotted in Fig. 15. The curves manifest that our online solver is much faster than the batch one. It is also observable that running time of the online solver grows linearly as the data scale increases, while the batch solver grows polynomially. Another phenomenon is that larger data actually work better by taking advantage under the online solver. For the medium size data, such as the Subway and Avenue datasets, online solver can still accelerate the process for 10–15 times. Notably, for large scale data, such as the day-level NYDP dataset, our online solver can achieve about 80 times speedup. It is certain that when the video is dozens or hundreds of hours long, the online solver will play an irreplaceable role and demonstrate its overwhelming advantage for constructing a practical system (Table 14).

6.10 Comparison with Other Sparsity Approaches

We compare testing efficiency with several other ℓ_1-norm minimization solvers surveyed in Yang et al. (2010). The methods we compare include gradient projection (GP) (Nowak et al. 2007) , homotopy (Osborne et al. 2000), ite. shrinkage- thresholding (IST) (Combettes and Wajs 2005), proximal gradient (PG) (Beck and Teboulle 2009), and augmented Lagrange multiplier (ALM) (Yang and Zhang 2011). The source codes are from Yang et al. (2010). Results in Table 15 show that our testing speed is much faster than ℓ_1-norm minimization. It is because our method only need to compute scores of matrix multiplication.

These sparsity methods also perform less well than ours in terms of distinguishing between normal and abnormal pat-

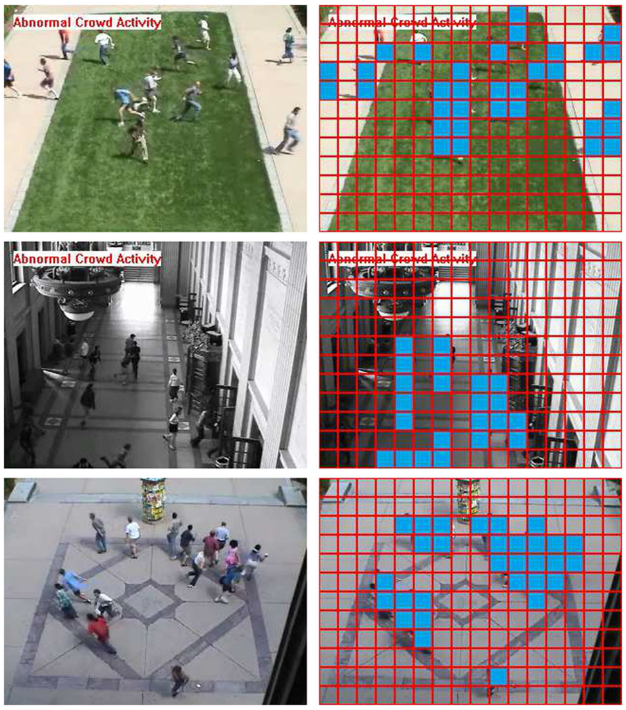

Fig. 13 Detected abnormal regions marked in blue (Color figure online)

terns. They select basis combination from a large basis pool where the reconstructed structure could deviate from input due to the enormous freedom. To verify it, we compare the average precision (AP) under different recalls in Table 15. Our approach is effective since the number of basis for combination is adaptive to information complexity.

6.11 Combining Autoencoding Feature

Our main contribution is a novel combination learning framework that is general to different video features. To demonstrate its generality, we use an autoencoding video feature proposed by Xu et al. (2015) to replace 3D gradient feature. Experiments in Tables 8, 9 and 11 show that the use of autoencoding feature advance the performance.

6.12 Separate Testing Cost Analysis

Our testing includes two main steps: feature extraction (3D cube gradient computing and PCA) and combination testing using Algorithm 2. Other minor procedures contain frame resizing, matrix reshape and so on. We list the average running time spent for each step to process one frame in the three datasets in Table 14.

6.13 Online Updating with Abnormality

We evaluate the robustness of our online solver toward abnormal samples. During testing, online update is preformed where learned combination from the training phase is taken as initialization. We use the UCSD Peb1 and sub-NYPD datasets where sub-NYPD is a 3-h non-dynamic

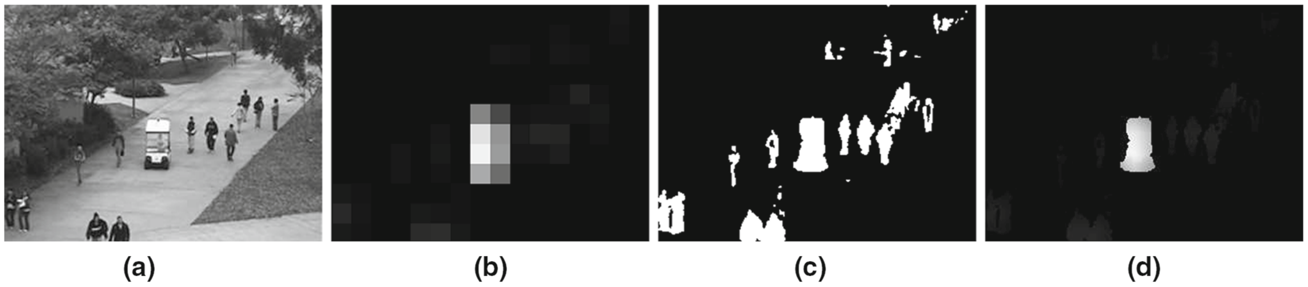

Fig. 14 Pixel-level process—the car is abnormal in this case. **a** Input frame, **b** Our regional detection map, **c** Foreground map, **d** Foreground bilateral filtering result

Table 10 The AUCs of different methods in the UCSD Ped2 dataset in frame level

	Adam et al. (2008)	Mehran et al. (2009)	MPPCA (Mahadevan et al. 2010)	MDT (Mahadevan et al. 2010)	Antic and Ommer (2011)	Xu et al. (2015)	SC (K = 10)	SC (K = 50)	SC (K = 11)	Ours (online)	Ours (batch)	Ours + Xu et al. (2015)
AUC	0.63	0.63	0.77	0.85	0.92	0.91	0.80	0.78	0.81	0.87	0.93	0.95

Subspace: replacing our combination learning by subspace clustering (Elhamifar and Vidal 2009) "Ours + Xu et al. (2015)" means replacing 3D gradient features by autoencoding feature proposed in Xu et al. (2015). SC (K = 10), (K = 50) and (K = 11) means subspace clustering method (Elhamifar and Vidal 2009) using subspace number 10, 50 and 11 respectively, where "K = 11" has best performance

Table 11 AUCs of different methods in the UCSD Ped2 dataset in pixel level

	Adam et al. (2008)	Mehran et al. (2009)	MPPCA (Mahadevan et al. 2010)	MDT (Mahadevan et al. 2010)	Antic and Ommer (2011)	SC (K = 10)	SC (K = 50)	SC (K = 13)	Kaltsa et al. (2015)	Li et al. (2014)	Ours (online)	(batch)	(FBLF)
AUC	0.18	0.14	0.21	0.44	0.76	0.59	0.58	0.60	0.75	0.78	0.64	0.67	0.80

Subspace: replacing our combination learning by subspace clustering (Elhamifar and Vidal 2009). “FBLF” means our abnormality maps are processed with foreground bilateral filter. C (K = 10), (K = 50) and (K = 13) means subspace clustering method (Elhamifar and Vidal 2009) using subspace number 10, 50 and 11 respectively, where “K = 11” has best performance

Table 12 AUC results in the UMN dataset

Methods	AUC (%)
Chaotic invariants (Wu et al. 2010)	99
Social force (Mehran et al. 2009)	96
Optical flow (Mehran et al. 2009)	84
Sparse (Cong et al. 2011)	97.5
Local aggregate (Saligrama and Chen 2012b)	98.5
Ours	98

sequence from NYPD. Results are shown in Table 16. It reveals that our batch solver is still better than the online ones since it captures globally normal patterns. Among the online solvers, testing with online update works slightly better. It is because extra normal patterns during testing contribute more information. In the meantime, abnormal events are rare among collected data and impose minor influence on combination update. We do not recommend the scheme of “testing with update” when running time is critical as it largely slows down computation.

7 Conclusion

We have presented an abnormal event detection method via *sparse combination learning*. This approach learns sparse combinations, which increase the testing speed hundreds to thousands of times without compromising effectiveness. An online solver is also provided to handle large-scale data. Our method achieves state-of-the-art results in several datasets. It is related to but differs largely from traditional subspace clus-

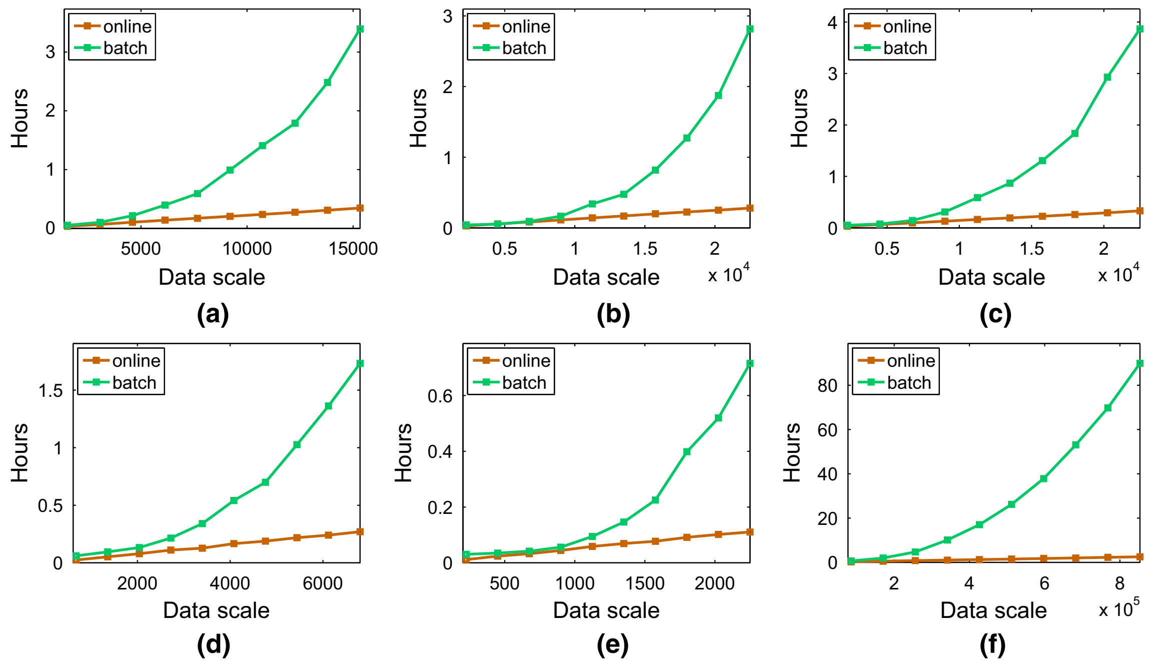

Fig. 15 Training cost as the data scale grows. **a** Avenue dataset, **b** subway entrance dataset, **c** subway exit dataset, **d** USCD (Ped1) dataset, **e** USCD (Ped2) dataset, **f** NYDP dataset

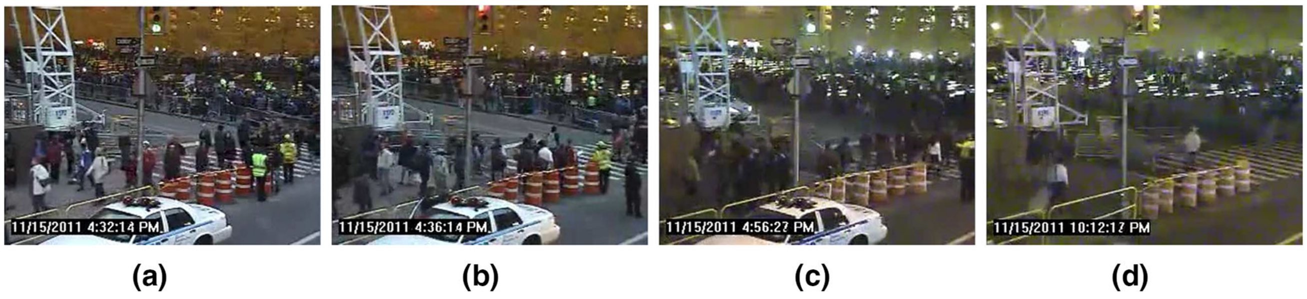

Fig. 16 Representative normal frames

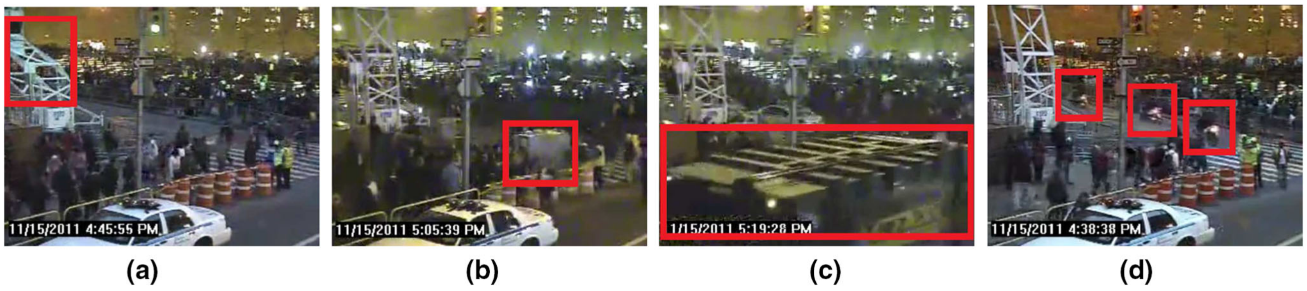

Fig. 17 Representative abnormal frames. **a** Steel bars fall, **b** a man pushes a big box, **c** a strange vehicle passes, and **d** three motorcycles pass

Table 13 Detection accuracy (in %) on the NYDP dataset

υ	0.2	0.3	0.4	0.5	0.6	0.7	0.8	Mean
Our (online, non-evolution)	47.4	43.4	40.6	36.1	32.5	27.9	16.7	34.9
Our (online,evolution)	50.2	46.7	43.5	39.4	35.2	30.3	18.1	37.62
Our (batch)	44.9	41.5	38.7	34.5	30.1	24.2	14.3	32.6
Subspaces	36.7	31.2	29.5	25.3	22.0	19.7	8.6	24.7

"evolution"("non-evolution") indicates dynamic patterns (stationary patterns) over time

Table 14 Average running time of processing one frame in each step on the four datasets

	Feature extraction (ms)	Combinations testing (ms)	Others (ms)	All (ms)	FPS
Avenue	0.6513	0.2641	0.1211	1.0365	964.8
UCSD Ped1	0.6126	0.2301	0.1132	0.9559	1046.1
Subway	0.6721	0.2256	0.0871	0.9848	1015.4
NYPD	0.6225	0.2047	0.1041	0.9313	1073.8

"ms" is short for millisecond

Table 15 Comparison with ℓ_1-norm sparsity solvers

	GP (Nowak et al. 2007)	homotopy (Osborne et al. 2000)	IST (Combettes and Wajs 2005)	PG (Beck and Teboulle 2009)	ALM (Yang and Zhang 2011)	Ours
Testing time (ms)	432.11	317.34	109.08	227.24	631.73	0.62
Average precision (AP)	0.89	0.89	0.90	0.88	0.87	0.94

The first row contains the average testing time for single regions. The second row is for average precision under different recalls. The dataset is the UCSD Ped1 Dataset. The number of dictionary atoms is 512. "ms" is short for millisecond. All experiments are conducted on the same PC with a 2.2GHz CPU and 8G RAM

Table 16 AUC (Area under precision-recall curve) on sub-NYPD and USCD Ped1 datasets

	Batch solver (%)	Online solver (%)	Online solver (testing with updating) (%)
sub-NYPD	35.5	32.4	33.1
USCD Ped1	91.8	89.1	89.6

"Online Solver (testing with update)" means we preform online update during testing

tering. We believe it will greatly benefit many applications, as well as fundamental understanding of video structures from a new perspective.

Acknowledgements This work is supported by the National Science Foundation China, under Grant 61772332,51675342,61133009 and by a Grant from the Research Grants Council of the Hong Kong SAR (Project No. 2150760).

References

Adam, A., Rivlin, E., Shimshoni, I., & Reinitz, D. (2008). Robust real-time unusual event detection using multiple fixed-location monitors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(3), 555–560.

Antic, B., & Ommer, B. (2011). Video parsing for abnormality detection. In *International conference on computer vision (ICCV)* (pp. 2415–2422).

Basharat, A., Gritai, A., & Shah, M. (2008). Learning object motion patterns for anomaly detection and improved object detection. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 1–8).

Beck, A., & Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, *2*(1), 183–202.

Benezeth, Y., Jodoin, P.-M., Saligrama, V., & Rosenberger, C. (2009). Abnormal events detection based on spatio-temporal co-occurences. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Bertsekas, D. P. (1999). *Nonlinear programming*. Belmont, MA: Athena Scientific.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, *41*(3), 15.

Cheng, K.-W., Chen, Y.-T., & Fang, W.-H. (2015). Video anomaly detection and localization using hierarchical feature representation and Gaussian process regression. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2909–2917).

Combettes, P. L., & Wajs, V. R. (2005). Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, *4*(4), 1168–1200.

Cong, Y., Yuan, J., & Liu, J. (2011). Sparse reconstruction costs for abnormal event detection. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 3449–3456).

Cui, X., Liu, Q., Gao, M., & Metaxas, D. N. (2011). Abnormal detection using interaction energy potentials. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 3161–3167).

Elhamifar, E., & Vidal, R. (2009). Sparse subspace clustering. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Elhamifar, E., & Vidal, R. (2013). Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(11), 2765–2781.

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381–395.

Jager, M., Knoll, C., & Hamprecht, F. A. (2008). Weakly supervised learning of a classifier for unusual event detection. *IEEE Transactions on Image Processing*, *17*(9), 1700–1708.

Jiang, F., Wu, Y., & Katsaggelos, A. K. (2007). Abnormal event detection from surveillance video by dynamic hierarchical clustering. In *ICIP* (pp. 145–148).

Jiang, F., Wu, Y., & Katsaggelos, A. K. (2008). Abnormal event detection based on trajectory clustering by 2-depth greedy search. In *IEEE international conference on acoustics, speech and signal processing* (pp. 2129–2132).

Jianga, F., Yuan, J., Tsaftarisa, S. A., & Katsaggelosa, A. K. (2011). Anomalous video event detection using spatiotemporal context. *Computer Vision and Image Understanding*, *115*(3), 323–333.

Jouseok, K., & Kyoungmu, L. (2012). A unified framework for event summarization and rare event detection. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 1266–1273).

Kaltsa, V., Briassouli, A., Kompatsiaris, I., Hadjileontiadis, L. J., & Strintzis, M. G. (2015). Swarm intelligence for detecting interesting events in crowded environments. *IEEE Transactions on Image Processing*, *24*(7), 2153–2166.

Kim, J., & Grauman, K. (2009). Observe locally, infer globally: a space-time MRF for detecting abnormal activities with incremental updates. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2921–2928).

Kratz, L., & Nishino, K. (2009). Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 1446–1453).

Li, W., Mahadevan, V., & Vasconcelos, N. (2014). Anomaly detection and localization in crowded scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(1), 18–22.

Lu, C., Shi, J., & Jia, J. (2013a). Abnormal event detection at 150 fps in matlab. In *International conference on computer vision (ICCV)*.

Lu, C., Shi, J., & Jia, J. (2013b). Online robust dictionary learning. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 415–422).

Mahadevan, V., Li, W., Bhalodia, V., & Vasconcelos, N. (2010). Anomaly detection in crowded scenes. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Mairal, J., Bach, F., Ponce, J., & Sapiro, G. (2010). Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, *11*, 19–60.

Ma, Y., Yang, A. Y., Derksen, H., & Fossum, R. (2008). Estimation of subspace arrangements with applications in modeling and segmenting mixed data. *SIAM Review*, *50*(3), 413–458.

Mehran, R., Oyama, A., & Shah, M. (2009). Abnormal crowd behavior detection using social force model. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Nowak, R. D, & Wright, S. J., et al. (2007). Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing*.

Osborne, M. R., Presnell, B., & Turlach, B. A. (2000). A new approach to variable selection in least squares problems. *IMA Journal of Numerical Analysis*, 20(3).

Peng, X, Zhang, L., & Yi, Z. (2013). Scalable sparse subspace clustering. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Pruteanu-Malinici, I., & Carin, L. (2008). Infinite hidden Markov models for unusual-event detection in video. *IEEE Transactions on Image Processing*, *17*(5), 811–822.

Saligrama, V., & Chen, Z. (2012a). Video anomaly detection based on local statistical aggregates. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2112–2119).

Saligrama, V., & Chen, Z. (2012b). Video anomaly detection based on local statistical aggregates. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2112–2119).

Shet, V. D., Harwood, D., & Davis, L. S. (2006). Multivalued default logic for identity maintenance in visual surveillance. In *European conference on computer vision (ECCV)*.

Shi, Y., Gao, Y., & Wang, R. (2010). Real-time abnormal event detection in complicated scenes. In *International conference on pattern recognition (ICPR)* (pp. 3653–3656).

Shi, J., Ren, X., Dai, G., Wang, J., & Zhang, Z. (2011). A non-convex relaxation approach to sparse dictionary learning. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 1809–1816).

Szabo, Z., Poczos, B., & Lorincz, A. (2011). Online group-structured dictionary learning. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2865–2872).

Trevor, H., Robert, T., & Friedman, J. H. (2001). The elements of statistical learning. New York: Springer.

Unusual Crowd Activity Dataset. http://mha.cs.umn.edu/Movies/Crowd-Activity-All.avi.

Vidal, R., Ma, Y., & Sastry, S. (2005). Generalized principal component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(12), 1945–1959.

Wang, X., Ma, X., & Grimson, E. (2007). Unsupervised activity perception by hierarchical bayesian models. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 1–8).

Wu, S., Moore, B. E., & Shah, M. (2010). Chaotic invariants of Lagrangian particle trajectories for anomaly detection in crowded scenes. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Xu, D., Ricci, E., Yan, Y., Song, J., & Sebe, N. (2015). Learning deep representations of appearance and motion for anomalous event detection. arXiv preprint arXiv:1510.01553.

Yang, J., & Zhang, Y. (2011). Alternating direction algorithms for \ell_1-problems in compressive sensing. *SIAM Journal on Scientific Computing*, *33*(1), 250–278.

Zhang, D., Gatica-Perez, D., Bengio, S., & McCowan, I. (2005). Semi-supervised adapted hmms for unusual event detection. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhao, B., Fei-Fei, L., & Xing, E. P. (2011). Online detection of unusual events in videos via dynamic sparse coding. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Zhong, H., Shi, J., & Visontai, M. (2004). Detecting unusual activity in video. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.