Abnormal Event Detection in Videos using Spatiotemporal Autoencoder

Yong Shean Chong yshean@1utar.my
Yong Haur Tay tayyh@utar.edu.my

Lee Kong Chian Faculty of Engineering Science,
Universiti Tunku Abdul Rahman, 43000 Kajang, Malaysia.

January 9, 2017

Abstract

We present an efficient method for detecting anomalies in videos. Recent applications of convolutional neural networks have shown promises of convolutional layers for object detection and recognition, especially in images. However, convolutional neural networks are supervised and require labels as learning signals. We propose a spatiotemporal architecture for anomaly detection in videos including crowded scenes. Our architecture includes two main components, one for spatial feature representation, and one for learning the temporal evolution of the spatial features. Experimental results on Avenue, Subway and UCSD benchmarks confirm that the detection accuracy of our method is comparable to state-of-the-art methods at a considerable speed of up to 140 fps.

1 Introduction

With the rapid growth of video data, there is an increasing need not only for recognition of objects and their behaviour, but in particular for detecting the rare, interesting occurrences of unusual objects or suspicious behaviour in the large body of ordinary data. Finding such abnormalities in videos is crucial for applications ranging from automatic quality control to visual surveillance.

Meaningful events that are of interest in long video sequences, such as surveillance footage, often have an extremely low probability of occurring. As such, manually detecting such events, or anomalies, is a very meticulous job that often requires more manpower than is generally available. This has prompted the need for automated detection and segmentation of sequences of interest. However, present technology requires an enormous amount of configuration efforts on each video stream prior to the deployment of the video analysis process, even with that, those events are based on some predefined heuristics, which makes the detection model difficult to generalize to different surveillance scenes.

Video data is challenging to represent and model due to its high dimensionality, noise, and a huge variety of events and interactions. Anomalies are also highly contextual, for example, running in a restaurant would be an anomaly,

but running at a park would be normal. Moreover, the definition of anomaly can be ambiguous and often vaguely defined. A person may think walking around on a subway platform is normal, but some may think it should be flagged as an anomaly since it could be suspicious. These challenges have made it difficult for machine learning methods to identify video patterns that produce anomalies in real-world applications.

There are many successful cases in the related field of action recognition [27, 7, 6, 18]. However, these methods only applicable to labelled video footages where events of interest are clearly defined and does not involve highly occluded scenes, such as crowded scenes. Furthermore, the cost of labelling every type of event is extremely high. Even so, it is not guaranteed to cover every past and future events. The recorded video footage is likely not long enough to capture all types of activities, especially abnormal activities which rarely or never occurred.

Recent effort on detecting anomalies by treating the task as a binary classification problem (normal and abnormal) [33] proved it being effective and accurate, but the practicality of such method is limited since footages of abnormal events are difficult to obtain due to its rarity. Therefore, many researchers have turned to models that can be trained using little to no supervision, including spatiotemporal features [13, 32], dictionary learning [31] and autoencoders [23]. Unlike supervised methods, these methods only require unlabelled video footages which contain little or no abnormal event, which are easy to obtain in real-world applications. A description of these methodologies and their limitations are discussed in the next section.

This paper presents a novel framework to represent video data by a set of general features, which are inferred automatically from a long video footage through a deep learning approach. Specifically, a deep neural network composed of a stack of convolutional autoencoders was used to process video frames in an unsupervised manner that captured spatial structures in the data, which, grouped together, compose the video representation. Then, this representation is fed into a stack of convolutional temporal autoencoders to learn the regular temporal patterns.

Our proposed method is domain free (i.e., not related to any specific task, no domain expert required), does not require any additional human effort, and can be easily applied to different scenes. To prove the effectiveness of the proposed method we apply the method to real-world datasets and show that our method consistently outperforms similar methods while maintaining a short running time.

1.1 Our Contributions

The main characteristics of our approach and also the contributions of this research are as follows:

- We wish to reduce the labor-intensive effort in feature engineering that results in a representation of the data that can support effective machine learning. This can be done by replacing low-level handcrafted features with learned hierarchical features. With the help of autoencoders, we are able to find representative features by learning from data instead of forming suitable features based on our knowledge.

- We replace traditional sparse coding methods with autoencoders. Unlike existing methods, there is no separation between extracting feature representation of videos and learning a model of features. In addition, by having multiple layers of hidden units in autoencoder, hierarchical feature learning can be achieved.

2 Related Work

Most of these abnormal instances are beforehand unknown, as this would require predicting all the ways something could happen out of the norm. It is therefore simply impossible to learn a model for all that is abnormal or irregular. But how can we find an anomaly without what to look for?

Since it is easier to get video data where the scene is normal in contrast to obtaining what is abnormal, we could focus on a setting where the training data contains only normal visual patterns. A popular approach adopted by researchers in this area is to first learn the normal patterns from the training videos, then anomalies are detected as events deviated from the normal patterns [13, 2, 32, 12]. The majority of the work on anomaly detection relies on the extraction of local features from videos, that are then used to train a normalcy model.

Trajectories have long been popular in video analysis and anomaly detection [34, 12, 20, 17]. A common characteristic of trajectory-based approaches is the deviation of nominal classes of object trajectories in a training phase, and the comparison of new test trajectories against the nominal classes in an evaluation phase. A statistically significant deviation from all classes indicates an anomaly. However, the accuracy of trajectory analysis relies heavily on tracking, which precise tracking still remains a significant challenge in computer vision, particularly in complex situations. Tracking-based approaches are suitable for scenes with few objects but are impractical for detecting abnormal patterns in a crowded or complex scene.

Non-tracking approaches that focus on spatiotemporal anomalies in videos also exist. These rely mainly on extracting and analyzing local low-level visual features, such as the histogram of oriented gradients [30], the histogram of oriented flows [11] and optical flow [21], by employing spatiotemporal video volumes (dense sampling or interest point selection) [3]. These local features are then grouped in clusters, i.e., bags of visual words (BOV), according to similarity metrics. Their popularity is due to their low computational cost, as well as their ability to focus on abnormal behaviour, even in extremely crowded scenes [10]. Another similar technique is sparse reconstruction [2, 32]. The fundamental underlying assumption of these methods is that any new feature representation of a normal/anomalous event can be approximately modeled as a (sparse) linear combination of feature representations (of previously observed events) in a trained dictionary. This assumes that all previously observed events are normal events.

However, since classical BOV approaches group similar volumes (summarize), they destroy all compositional information in the process of grouping visual words. It is also required to pre-determine the number of clusters, which can only be found through trial-and-error during testing time. In addition, codebook models require searching over a large space [22] even during the time

of testing, making it impractical for real-time anomaly detection.

The success of deep learning methods in various applications consequently caused the rise of such methods in anomaly detection. The term deep learning refers to learning a hierarchical set of features through multiple layers of hidden nodes in an artificial neural network. Unlike previously stated methods, there is no need to define a specific set of features to extract from the dataset – deep learning methods learn the useful features directly from the data with minimal preprocessing. Specifically, convolutional neural networks (ConvNet) have proved its effectiveness in a wide range of applications such as object recognition [26], person detection [28], and action recognition [27, 25]. ConvNet consists of a stack of convolutional layers with a fully-connected layer and a softmax classifier, and convolutional autoencoder is essentially a ConvNet with its fully-connected layer and classifier replaced by a mirrored stack of convolutional layers. The authors of [33] applied a 3D ConvNet on classifying anomalies, whereas [5] used an end-to-end convolutional autoencoder to detect anomalies in surveillance videos. Their reported result proves the usefulness of learned representation on videos through a stack of convolutional layers. On the other hand, long short term memory (LSTM) model is well-known for learning temporal patterns and predicting time series data. [15] has recently proposed to apply convolutional LSTMs for learning the regular temporal patterns in videos and his findings show great promise of what deep neural network can learn.

Despite its simplicity, some limitations remain in these recently proposed methods. Though 3D ConvNet performed excellently in learning discriminative features between the anomalies and the normal events, it is impractical to apply in real-world scenarios due to the absence of video segments containing abnormal events. Meanwhile, in the convolutional autoencoder proposed by [5], convolution and pooling operations are performed only spatially, even though the proposed network takes multiple frames as input, because of the 2D convolutions, after the first convolution layer, temporal information is collapsed completely [27]. Besides, convolutional LSTM layers applied by [15] are memory-intensive – the training will need to be executed on very small mini-batches, which results in slow training and testing time.

3 Methodology

The method described here is based on the principle that when an abnormal event occurs, the most recent frames of video will be significantly different than the older frames. Inspired by [5], we train an end-to-end model that consists of a spatial feature extractor and a temporal encoder-decoder which together learns the temporal patterns of the input volume of frames. The model is trained with video volumes consists of only normal scenes, with the objective to minimize the reconstruction error between the input video volume and the output video volume reconstructed by the learned model. After the model is properly trained, normal video volume is expected to have low reconstruction error, whereas video volume consisting of abnormal scenes is expected to have high reconstruction error. By thresholding on the error produced by each testing input volumes, our system will be able to detect when an abnormal event occurs.

Our approach consists of three main stages:

3.1 Preprocessing

The task of this stage is to convert raw data to the aligned and acceptable input for the model. Each frame is extracted from the raw videos and resized to 227×227. To ensure that the input images are all on the same scale, the pixel values are scaled between 0 and 1 and subtracted every frame from its global mean image for normalization. The mean image is calculated by averaging the pixel values at each location of every frame in the training dataset. After that, the images are converted to grayscale to reduce dimensionality. The processed images are then normalized to have zero mean and unit variance.

The input to the model is video volumes, where each volume consists of 10 consecutive frames with various skipping strides. As the number of parameters in this model is large, large amount of training data is needed. Following [5]s practice, we perform data augmentation in the temporal dimension to increase the size of the training dataset. To generate these volumes, we concatenate frames with stride-1, stride-2, and stride-3. For example, the first stride-1 sequence is made up of frame {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}, whereas the first stride-2 sequence contains frame number {1, 3, 5, 7, 9, 11, 13, 15, 17, 19}, and stride-3 sequence would contain frame number {1, 4, 7, 10, 13, 16, 19, 22, 25, 28}. Now the input is ready for model training.

3.2 Feature Learning

We propose a convolutional spatiotemporal autoencoder to learn the regular patterns in the training videos. Our proposed architecture consists of two parts — spatial autoencoder for learning spatial structures of each video frame, and temporal encoder-decoder for learning temporal patterns of the encoded spatial structures. As illustrated in Figure 1 and 2, the spatial encoder and decoder have two convolutional and deconvolutional layers respectively, while the temporal encoder is a three-layer convolutional long short term memory (LSTM) model. Convolutional layers are well-known for its superb performance in object recognition, while LSTM model is widely used for sequence learning and time-series modelling and has proved its performance in applications such as speech translation and handwriting recognition.

3.2.1 Autoencoder

Autoencoders, as the name suggests, consist of two stages: encoding and decoding. It was first used to reduce dimensionality by setting the number of encoder output units less than the input. The model is usually trained using back-propagation in an unsupervised manner, by minimizing the reconstruction error of the decoding results from the original inputs. With the activation function chosen to be nonlinear, an autoencoder can extract more useful features than some common linear transformation methods such as PCA.

3.2.2 Spatial Convolution

The primary purpose of convolution in case of a convolutional network is to extract features from the input image. Convolution preserves the spatial relationship between pixels by learning image features using small squares of input data. Mathematically, convolution operation performs dot products between the

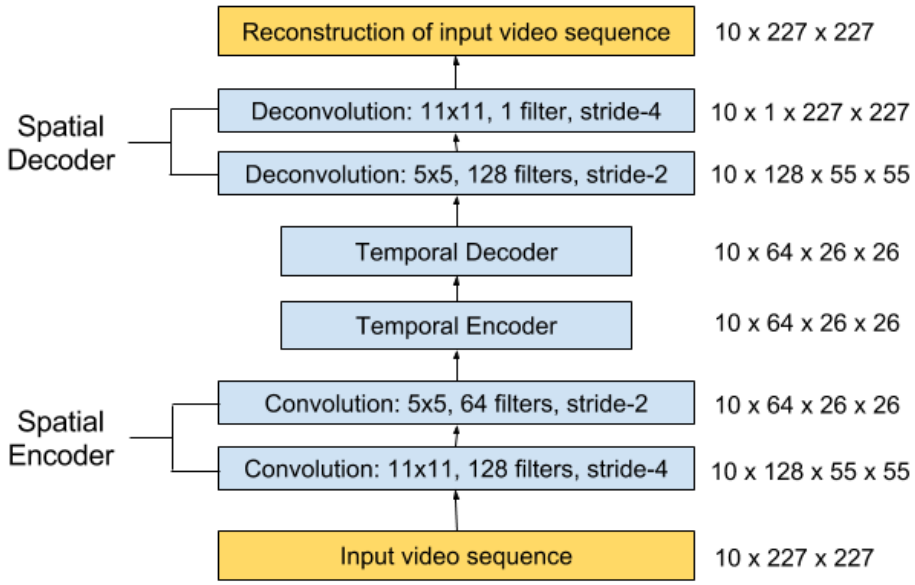

Figure 1: Our proposed network architecture. It takes a sequence of length T as input, and output a reconstruction of the input sequence. The numbers at the rightmost denote the output size of each layer. The spatial encoder takes one frame at a time as input, after which $T = 10$ frames have been processed, the encoded features of 10 frames are concatenated and fed into temporal encoder for motion encoding. The decoders mirror the encoders to reconstruct the video volume.

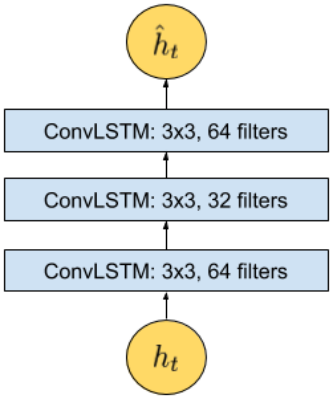

Figure 2: The zoomed-in architecture at time t, where t is the input vector at this time step. The temporal encoder-decoder model has 3 convolutional LSTM (ConvLSTM) layers.

filters and local regions of the input. Suppose that we have some $n \times n$ square input layer which is followed by the convolutional layer. If we use an $m \times m$ filter W, the convolutional layer output will be of size $(n-m+1) \times (n-m+1)$.

A convolutional network learns the values of these filters on its own during the training process, although we still need to specify parameters such as the number of filters, filter size, the number of layers before training. With more number of filters we have, more image features get extracted and the better the network becomes at recognizing patterns in unseen images. However, more filters would add to computational time and exhaust memory faster, so we need to find balance by not setting the number of filters too large.

3.2.3 Recurrent Neural Network (RNN)

In a traditional feedforward neural network, we assume that all inputs (and outputs) are independent of each other. However, learning temporal dependencies between inputs are important in tasks involving sequences, for example, a word predictor model should be able to derive information from the past inputs. RNN works just like a feedforward network, except that the values of its output vector are influenced not only by the input vector but also on the entire history of inputs. In theory, RNNs can make use of information in arbitrarily long sequences, but in practice, they are limited to looking back only a few steps due to vanishing gradients.

3.2.4 Long Short Term Memory (LSTM)

To overcome this problem, a variant of RNN is introduced: long short term memory (LSTM) model which incorporates a recurrent gate called forget gate. With the new structure, LSTMs prevent backpropagated errors from vanishing or exploding, thus can work on long sequences and they can be stacked together to capture higher level information. The formulation of a typical LSTM unit is summarized with Figure 3 and equations (1) through (6).

$$f_t = \sigma(W_f \otimes [h_{t-1}, x_t] + b_f) \tag{1}$$

$$i_t = \sigma(W_i \otimes [h_{t-1}, x_t] + b_i) \tag{2}$$

$$\hat{C}_t = tanh(W_C \otimes [h_{t-1}, x_t] + b_C) \tag{3}$$

$$C_t = f_t \otimes C_{t-1} + i_t \otimes \hat{C}_t \tag{4}$$

$$o_t = \sigma(W_o \otimes [h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t \otimes tanh(C_t) \tag{6}$$

Equation (1) represents the forget layer, (2) and (3) are where new information is added, (4) combines old and new information, whereas (5) and (6) output what has been learned so far to the LSTM unit at the next timestep. The variable x_t denotes the input vector, h_t denotes the hidden state, and C_t denotes the cell state at time t. W are the trainable weight matrices, b are the bias vectors, and the symbol $\otimes$ denotes the Hadamard product.

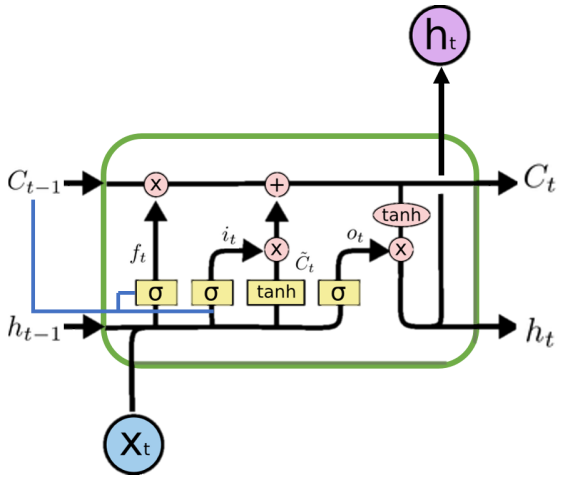

Figure 3: The structure of a typical LSTM unit. The blue line represents an optional peephole structure, which allows the internal state to look back (peep) at the previous cell state C_{t-1} for a better decision. Best viewed in colour.

3.2.5 Convolutional LSTM

A variant of the LSTM architecture, namely Convolutional Long Short-term Memory (ConvLSTM) model was introduced by Shi et al. in [24] and has been recently utilized by Patraucean et al. in [19] for video frame prediction. Compared to the usual fully connected LSTM (FC-LSTM), ConvLSTM has its matrix operations replaced with convolutions. By using convolution for both input-to-hidden and hidden-to-hidden connections, ConvLSTM requires fewer weights and yield better spatial feature maps. The formulation of the ConvLSTM unit can be summarized with (7) through (12).

$$f_t = \sigma(W_f * [h_{t-1}, x_t, C_{t-1}] + b_f) \tag{7}$$

$$i_t = \sigma(W_i * [h_{t-1}, x_t, C_{t-1}] + b_i) \tag{8}$$

$$\hat{C}_t = tanh(W_C * [h_{t-1}, x_t] + b_C) \tag{9}$$

$$C_t = f_t \otimes C_{t-1} + i_t \otimes \hat{C}_t \tag{10}$$

$$o_t = \sigma(W_o * [h_{t-1}, x_t, C_{t-1}] + b_o) \tag{11}$$

$$h_t = o_t \otimes tanh(C_t) \tag{12}$$

While the equations are similar in nature to (1) through (6), the input is fed in as images, while the set of weights for every connection is replaced by convolutional filters (the symbol $*$ denotes a convolution operation). This allows ConvLSTM work better with images than the FC-LSTM due to its ability to propagate spatial characteristics temporally through each ConvLSTM state.

Note that this convolutional variant also adds an optional 'peephole' connections to allow the unit to derive past information better.

3.3 Regularity Score

Once the model is trained, we can evaluate our models performance by feeding in testing data and check whether it is capable of detecting abnormal events while keeping false alarm rate low. To better compare with [5], we used the same formula to calculate the regularity score for all frames, the only difference being the learned model is of a different kind. The reconstruction error of all pixel values I in frame t of the video sequence is taken as the Euclidean distance between the input frame and the reconstructed frame:

$$e(t) = ||x(t) - f_W(x(t))||_2 \tag{13}$$

where f_W is the learned weights by the spatiotemporal model. We then compute the abnormality score $s_a(t)$ by scaling between 0 and 1. Subsequently, regularity score $s_r(t)$ can be simply derived by subtracting abnormality score from 1:

$$s_a(t) = \frac{e(t) - e(t)_{min}}{e(t)_{max}} \tag{14}$$

$$s_r(t) = 1 - s_a(t) \tag{15}$$

3.4 Anomaly Detection

3.4.1 Thresholding

It is straightforward to determine whether a video frame is normal or anomalous. The reconstruction error of each frame determines whether the frame is classified as anomalous. The threshold determines how sensitive we wish the detection system to behave — for example, setting a low threshold makes the system become sensitive to the happenings in the scene, where more alarms would be triggered. We obtain the true positive and false positive rate by setting at different error threshold in order to calculate the area under the receiver operating characteristic (ROC) curve (AUC). The equal error rate (EER) is obtained when false positive rate equals to the false negative rate.

3.4.2 Event count

Following the practice in [5], to reduce the noisy and unmeaningful minima in the regularity score, we applied Persistence1D [9] algorithm to group local minima with a fixed temporal window of 50 frames. We assume local minima within 50 frames belong to the same abnormal event. This is a reasonable length of the temporal window as an abnormal event should be at least 2-3 seconds long to be meaningful (videos are captured at 24-25 fps).

4 Experiments

4.1 Datasets

We train our model on five most commonly used benchmarking datasets: Avenue [13], UCSD Ped1 and Ped2 [14], Subway entrance and exit datasets [1]. All videos are taken from a fixed position for each dataset. All training videos contain only normal events. Testing videos have both normal and abnormal events.

In Avenue dataset, there are total 16 training and 21 testing video clips. Each clips duration vary between less than a minute to two minutes long. The normal scenes consist of people walking between staircase and subway entrance, whereas the abnormal events are people running, walking in opposite direction, loitering and etc. The challenges of this dataset include camera shakes and a few outliers in the training data. Also, some normal pattern seldom appears in the training data.

UCSD Ped1 dataset has 34 training and 36 testing video clips, where each clip contains 200 frames. The videos consist of groups of people walking towards and away from the camera. UCSD Ped2 dataset has 16 training and 12 testing video clips, where the number of frames of each clip varies. The videos consist of walking pedestrians parallel to the camera plane. Anomalies of the two datasets include bikers, skaters, carts, wheelchairs and people walking in the grass area.

Subway entrance dataset is 1 hour 36 minutes long with 66 unusual events of five different types: walking in the wrong direction (WD), no payment (NP), loitering (LT), irregular interactions between people (II), and miscellaneous (e.g. sudden stop, running fast). First 20 minutes of the video is used for training.

Subway exit dataset is 43 minutes long with 19 unusual events of three types: walking in the wrong direction (WD), loitering (LT), and miscellaneous (e.g. sudden stop, looking around, a janitor cleaning the wall, gets off the train and gets on the train again quickly. First 5 minutes of the video is used for training.

4.2 Model Parameters

We train the model by minimizing the reconstruction error of the input volume. We use Adam optimizer to allow it taking the role of setting the learning rate automatically based on the models weight update history. We use mini-batches of size 64 and each training volume is trained for a maximum of 50 epochs or until the reconstruction loss of validation data stop decreasing after 10 consecutive epochs. Hyperbolic tangent is chosen as the activation function of spatial encoder and decoder. To ensure the symmetry of the encoding and decoding function, we did not use rectified linear unit (ReLU) despite its regularization ability because activated values from ReLU have no upper bound.

4.3 Results and Analysis

4.3.1 Quantitative Analysis: ROC and Anomalous Event Count

Table 1 shows the frame-level AUC and EER of our and of other methods on all five datasets. We outperform all other considered methods in respect to frame-level EER. We also provide the event count comparison for Avenue dataset and

Table 1: Comparison of area under ROC curve (AUC) and Equal Error Rate (EER) of different methods. Higher AUC and lower EER are better. Most papers did not publish their AUC/EER for avenue, subway entrance and exit dataset.

Method	AUC/EER (%)				
	Ped1	Ped2	Avenue	Subway Entrance	Subway Exit
Adam [1]	77.1/38.0	-/42.0			
SF [16]	67.5/31.0	55.6/42.0		N/A	
MPPCA [14]	66.8/40.0	69.3/30.0			
MPPCA+SF [14]	74.2/32.0	61.3/36.0			
HOFME [29]	72.7/33.1	87.5/20.0	N/A	81.6/**22.8**	84.9/17.8
ConvAE [5]	81.0/27.9	**90.0**/21.7	70.2/25.1	**94.3**/26.0	80.7/9.9
Ours	**89.9**/**12.5**	87.4/**12.0**	**80.3**/**20.7**	84.7/23.7	**94.0**/**9.5**

Table 2: Anomalous event and false alarm count detected by different methods. GT denotes groundtruth values of event count.

Method	Anomalous Event Detected / False Alarm		
	Avenue (GT: 47, smaller set GT: 14)	Subway Entrance (GT: 66)	Subway Exit (GT: 19)
Sparse combination [13]	12/1 (smaller set)	57/4	19/2
Space-time MRF [8]	N/A	56/3	18/0
Online [4]	N/A	60/5	19/2
ConvAE [5]	45/4	61/15	17/5
Ours	44/6	61/9	18/10

the entrance and exit scenes in the Subway dataset in Table 2. For the entrance scenes, we are better than [5] since we detect the same number of anomalies with less false alarms. For the exit scenes, we detected more abnormal events compared to [5] but at the expense of higher false alarm rate.

The event count breakdown according to type of event is presented in Table 3, 4 and 5 for Avenue dataset, Subway entrance and exit datasets respectively. All throwing, loitering (LT) and irregular interaction (II) events are well captured by our proposed system. These are strong abnormalities that are significantly different from what was captured in the normal scenes. However, our system does have difficulties in detecting certain types of event. Missed detection of running and walking in opposite direction events are due to (1) the crowded activities where multiple foreground events take place; and (2) the object of interest is far away from the camera. Meanwhile, in Subway entrance and exit scenes, some wrong direction events are missed. On the other hand, some no payment (NP) events in Subway entrance scene are difficult to detect due to their similar motion compared to others walking through the barrier.

We also present a run-time analysis on our proposed abnormal event detec-

Table 3: Anomalous event and false alarm count detected by different methods on various event type in Avenue dataset.

	Run	Loiter	Throw	Opposite Direction	False Alarm
Groundtruth	12	8	19	8	0
Ours	10	8	19	7	12

Table 4: Anomalous event and false alarm count detected by different methods on various event type in Subway Entrance dataset. WD: wrong direction; NP: no payment; LT: loitering; II: irregular interaction; Misc.: miscellaneous.

	WD	NP	LT	II	Misc.	False Alarm
Groundtruth	26	13	14	4	9	0
Ours	24	10	14	4	9	9

Table 5: Anomalous event and false alarm count detected by different methods on various event type in Subway Exit dataset. WD: wrong direction; LT: loitering; Misc.: miscellaneous.

	WD	LT	Misc.	False Alarm
Groundtruth	9	3	7	0
Ours	8	3	7	10

Table 6: Details of run-time during testing (second/frame).

	Time (in sec)			
	Preprocessing	Representation	Classifying	Total
CPU	0.0010	0.2015	0.0002	0.2027 (∼5fps)
GPU	0.0010	0.0058	0.0002	0.0070 (∼143fps)

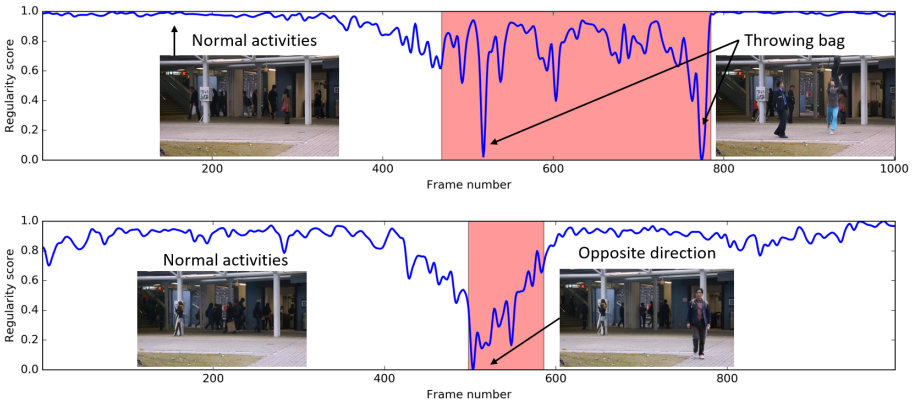

Figure 4: Regularity score of video #5 (top) and #15 (bottom) from the Avenue dataset.

tion system, on CPU (Intel Xeon E5-2620) and GPU (NVIDIA Maxwell Titan X) respectively, in Table 6. The total time taken is well less than a quarter second per frame for both CPU and GPU configuration. Due to computational intensive multiplication operations when feeding the input through the convolutional autoencoders, it is recommended to run on GPU for a better speed of nearly 30 times faster than CPU.

4.3.2 Qualitative Analysis: Visualising Frame Regularity

Figure 4, 5, and 6 illustrate the output of the proposed system on samples of the Avenue dataset, Subway entrance and exit scenes respectively; our method detects anomalies correctly in these cases even in crowded scenes.

Almost all anomalies produce strong downward spikes which indicate a low regularity score, including a difficult-to-detect skateboarding activity as illus-

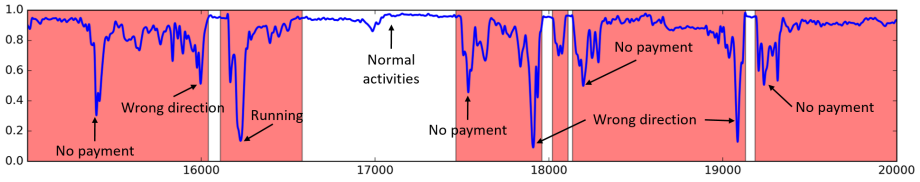

Figure 5: Regularity score of frames 115000-120000 from the Subway Entrance video.

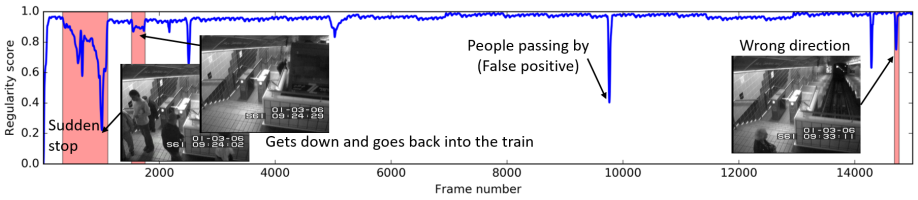

Figure 6: Regularity score of frames 22500-37500 from the Subway Entrance video.

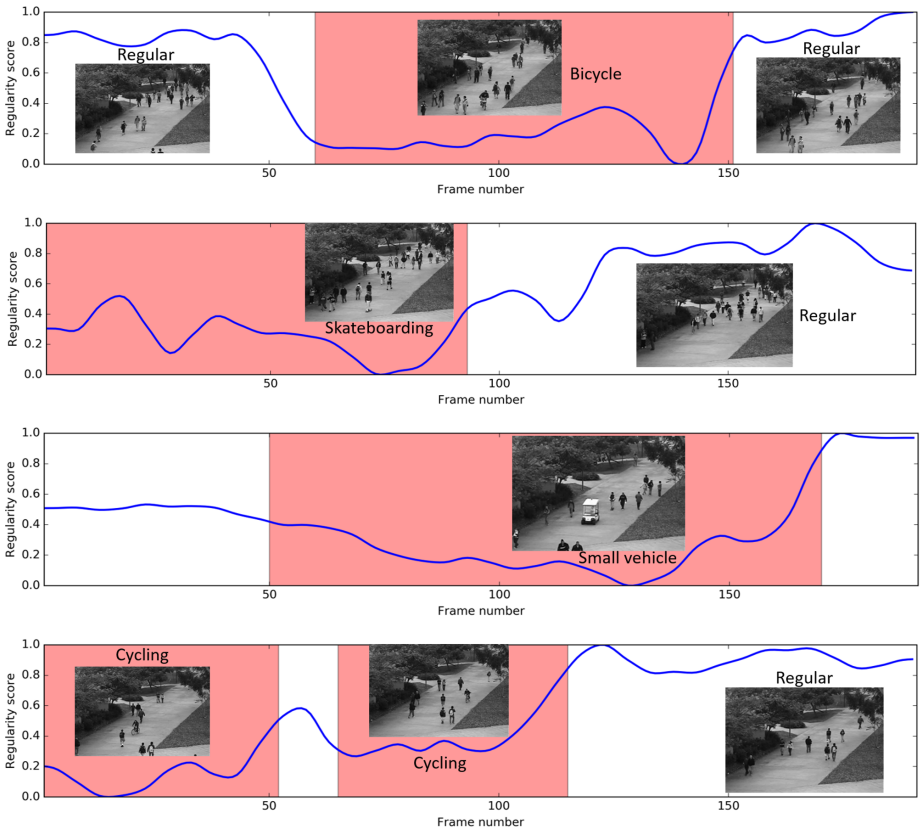

Figure 7: Regularity score of video #1, #8, #24 and #32 (from top to bottom) from UCSD Ped1 dataset.

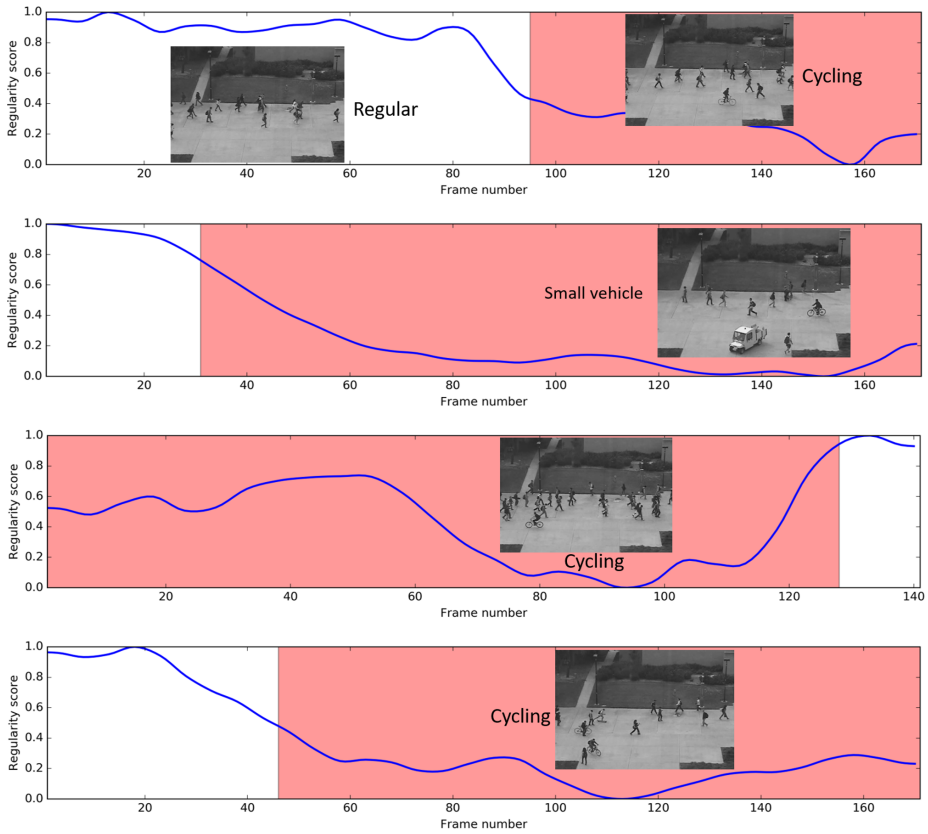

Figure 8: Regularity score of video #2, #4, #5 and #7 (from top to bottom) from UCSD Ped2 dataset.

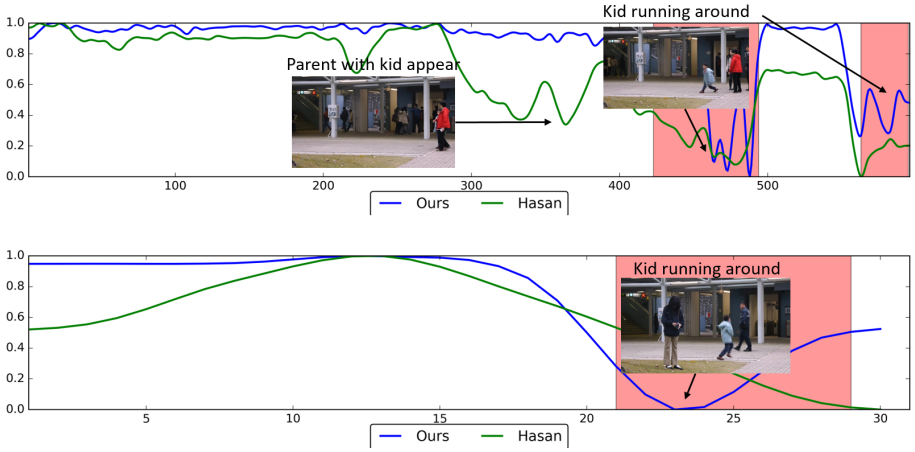

Figure 9: Comparing our method with ConvAE [5] on Avenue dataset video #7 (top) and #8 (bottom). Best viewed in colour.

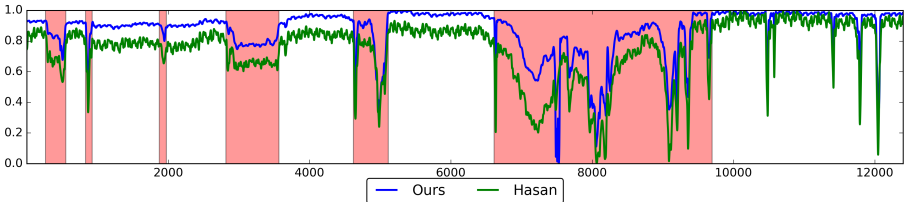

Figure 10: Comparing our method with ConvAE [5] on Subway Exit video frames 10000-22500. Best viewed in colour.

trated in Figure 7.

4.3.3 Comparing Our Method with 2D Convolutional Autoencoder (ConvAE)

From Figure 9 and 10, it is easy to see that our method has detected more abnormal events with fewer false alarms compared to [5]. As observed in Figure 11, our method is able to produce higher regularity score during normal activities and lower scores when there are abnormalities.

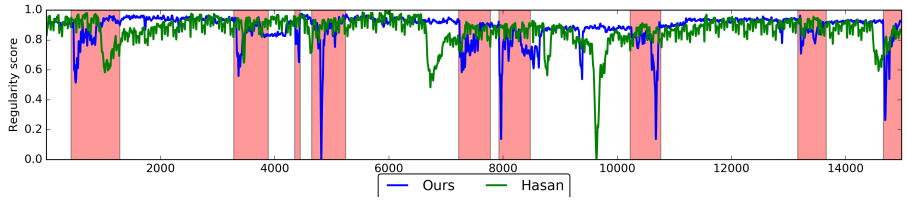

Figure 11: Comparing our method with ConvAE [5] on Subway Entrance video frames 120000-144000. Best viewed in colour.

5 Conclusion

In this research, we have successfully applied deep learning to the challenging video anomaly detection problem. We formulate anomaly detection as a spatiotemporal sequence outlier detection problem and applied a combination of spatial feature extractor and temporal sequencer ConvLSTM to tackle the problem. The ConvLSTM layer not only preserves the advantages of FC-LSTM but is also suitable for spatiotemporal data due to its inherent convolutional structure. By incorporating convolutional feature extractor in both spatial and temporal space into the encoding-decoding structure, we build an end-to-end trainable model for video anomaly detection. The advantage of our model is that it is semi-supervised – the only ingredient required is a long video segment containing only normal events in a fixed view. Despite the models ability to detect abnormal events and its robustness to noise, depending on the activity complexity in the scene, it may produce more false alarms compared to other methods. For future work, we will investigate how to improve the result of video anomaly detection by active learning – having human feedback to update the learned model for better detection and reduced false alarms. One idea is to add a supervised module to the current system, which the supervised module works only on the video segments filtered by our proposed method, then train a discriminative model to classify anomalies when enough video data has been acquired.

References

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

[2] Cong, Y., Yuan, J., Liu, J.: Sparse reconstruction cost for abnormal event detection. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 3449–3456 (2011)

[3] Dollár, P., Rabaud, V., Cottrell, G., Belongie, S.: Behavior recognition via sparse spatio-temporal features. In: Proceedings - 2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, VS-PETS. vol. 2005, pp. 65–72 (2005)

[4] Dutta, J., Banerjee, B.: Online detection of abnormal events using incremental coding length (2015), http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9923

[5] Hasan, M., Choi, J., Neumann, J., Roy-Chowdhury, A.K., Davis, L.S.: Learning temporal regularity in video sequences. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 733–742 (June 2016)

[6] Ji, S., Yang, M., Yu, K.: 3D Convolutional Neural Networks for Human Action Recognition. Pami 35(1), 221–31 (2013), http://www.ncbi.nlm.nih.gov/pubmed/22392705

[7] Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1725–1732 (June 2014)

[8] Kim, J., Grauman, K.: Observe locally, infer globally: A space-time MRF for detecting abnormal activities with incremental updates. In: 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009. pp. 2921–2928 (2009)

[9] Kozlov, Y., Weinkauf, T.: Persistence1d: Extracting and filtering minima and maxima of 1d functions. `http://people.mpi-inf.mpg.de/~weinkauf/notes/persistence1d.html`, accessed: 2017-01-05

[10] Kratz, L., Nishino, K.: Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models. In: 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009. pp. 1446–1453 (2009)

[11] Laptev, I., Marszałek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies. In: 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR (2008)

[12] Li, C., Han, Z., Ye, Q., Jiao, J.: Abnormal behavior detection via sparse reconstruction analysis of trajectory. In: Proceedings - 6th International Conference on Image and Graphics, ICIG 2011. pp. 807–810 (2011)

[13] Lu, C., Shi, J., Jia, J.: Abnormal event detection at 150 fps in matlab. In: 2013 IEEE International Conference on Computer Vision. pp. 2720–2727 (Dec 2013)

[14] Mahadevan, V., Li, W., Bhalodia, V., Vasconcelos, N.: Anomaly detection in crowded scenes. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 1975–1981 (2010)

[15] Medel, J.R.: Anomaly Detection Using Predictive Convolutional Long Short-Term Memory Units. Master's thesis, Rochester Institute of Technology (2016), accessed from http://scholarworks.rit.edu/theses/9319

[16] Mehran, R., Oyama, A., Shah, M.: Abnormal crowd behavior detection using social force model. In: 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009. pp. 935–942 (2009)

[17] Mo, X., Monga, V., Bala, R., Fan, Z.: Adaptive sparse representations for video anomaly detection. IEEE Transactions on Circuits and Systems for Video Technology 24(4), 631–645 (2014)

[18] Oneata, D., Verbeek, J., Schmid, C.: Action and Event Recognition with Fisher Vectors on a Compact Feature Set. 2013 IEEE International Conference on Computer Vision pp. 1817–1824 (2013), `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6751336`

[19] Patraucean, V., Handa, A., Cipolla, R.: Spatio-temporal video autoencoder with differentiable memory. International Conference On Learning Representations (2015), 1–10 (2016), http://arxiv.org/abs/1511.06309

[20] Piciarelli, C., Micheloni, C., Foresti, G.L.: Trajectory-based anomalous event detection. IEEE Transactions on Circuits and Systems for Video Technology 18(11), 1544–1554 (2008)

[21] Reddy, V., Sanderson, C., Lovell, B.C.: Improved anomaly detection in crowded scenes via cell-based analysis of foreground speed, size and texture. In: CVPR 2011 WORKSHOPS. pp. 55–61 (June 2011)

[22] Roshtkhari, M.J., Levine, M.D.: An on-line, real-time learning method for detecting anomalies in videos using spatio-temporal compositions. Computer Vision and Image Understanding 117(10), 1436–1452 (2013)

[23] Sabokrou, M., Fathy, M., Hoseini, M., Klette, R.: Real-time anomaly detection and localization in crowded scenes. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 56–62 (June 2015)

[24] Shi, X., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.k., Woo, W.c.: Convolutional lstm network: A machine learning approach for precipitation nowcasting. In: Proceedings of the 28th International Conference on Neural Information Processing Systems. pp. 802–810. NIPS'15, MIT Press, Cambridge, MA, USA (2015), http://dl.acm.org/citation.cfm?id=2969239.2969329

[25] Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Advances in Neural Information Processing Systems. pp. 568–576 (2014)

[26] Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-Scale Image Recognition. ImageNet Challenge pp. 1–10 (2014), http://arxiv.org/abs/1409.1556

[27] Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: 2015 IEEE International Conference on Computer Vision (ICCV). pp. 4489–4497 (Dec 2015)

[28] Vu, T.H., Osokin, A., Laptev, I.: Context-aware cnns for person head detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2893–2901 (2015)

[29] Wang, T., Snoussi, H.: Histograms of optical flow orientation for abnormal events detection. In: IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, PETS. pp. 45–52 (2013)

[30] Xiao, T., Zhang, C., Zha, H., Wei, F.: Anomaly detection via local coordinate factorization and spatio-temporal pyramid. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). vol. 9007, pp. 66–82 (2015)

[31] Yen, S.H., Wang, C.H.: Abnormal event detection using HOSF. In: 2013 International Conference on IT Convergence and Security, ICITCS 2013 (2013)

[32] Zhao, B., Fei-Fei, L., Xing, E.P.: Online detection of unusual events in videos via dynamic sparse coding. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 3313–3320 (2011)

[33] Zhou, S., Shen, W., Zeng, D., Fang, M., Wei, Y., Zhang, Z.: Spatial-temporal convolutional neural networks for anomaly detection and localization in crowded scenes. Signal Processing: Image Communication 47, 358–368 (sep 2016)

[34] Zhou, S., Shen, W., Zeng, D., Zhang, Z.: Unusual event detection in crowded scenes by trajectory analysis. In: ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings. vol. 2015-Augus, pp. 1300–1304. Institute of Electrical and Electronics Engineers Inc. (Aug 2015)