Multi-timescale Trajectory Prediction for Abnormal Human Activity Detection

Royston Rodrigues
NEC Corporation, Japan
r-rodrigues@nec.com

Neha Bhargava
Oxford Brookes University
neha.iitb@gmail.com

Rajbabu Velmurugan
Indian Institute of Technology Bombay
rajbabu@ee.iitb.ac.in

Subhasis Chaudhuri
Indian Institute of Technology Bombay
sc@ee.iitb.ac.in

Abstract

*A classical approach to abnormal activity detection is to learn a representation for normal activities from the training data and then use this learned representation to detect abnormal activities while testing. Typically, the methods based on this approach operate at a fixed timescale — either a single time-instant (*e.g. *frame-based) or a constant time duration (*e.g. *video-clip based). But human abnormal activities can take place at different timescales. For example, jumping is a short-term anomaly and loitering is a long-term anomaly in a surveillance scenario. A single and pre-defined timescale is not enough to capture the wide range of anomalies occurring with different time duration. In this paper, we propose a multi-timescale model to capture the temporal dynamics at different timescales. In particular, the proposed model makes future and past predictions at different timescales for a given input pose trajectory. The model is multi-layered where intermediate layers are responsible to generate predictions corresponding to different timescales. These predictions are combined to detect abnormal activities. In addition, we also introduce a single-camera abnormal activity dataset for research use that contains 483,566 annotated frames. Our experiments show that the proposed model can capture the anomalies of different time duration and outperforms existing methods.*

1. Introduction

Detecting abnormal activities is a challenging problem in computer vision. There is no generic definition available for abnormal events and is usually dependent on the scene under consideration. For example, *cycling* on a *footpath* is typically an abnormal activity whereas it becomes normal on a *road*. To address such a scene context dependency, a typical approach is to consider rare or unseen events in a scene as abnormal. But this may classify the unseen normal activities as abnormal. In general, it may not be possible to know all the normal and abnormal activities during training. We only have access to subsets of normal and abnormal activities. The lack of a generic definition and insufficiency in the data, make it extremely hard for any learning algorithm to understand and capture the nature of abnormal activity.

More often, the abnormal activity detection problem is posed as an unsupervised learning problem. A common setup of the problem is this - the training data consists of only normal activities and the test data contains normal as well as abnormal activities. A standard approach is to build a model that captures the normality present in the training data. During testing, any deviation from the learned normality indicates the level of abnormality in the test data. Many existing methods formulate it as an outlier detection problem [5, 19, 17, 12, 15]. They attempt to fit the features corresponding to normal activities in a hyper-sphere and the distance of a test feature from this hyper-sphere indicates its abnormality.

One major limitation of the current methods is that they are trained at a fixed timescale - either a single time-step or a constant number of time-steps. This restricts the model to build understanding of training data at that timescale and hence it may not capture anomalies that occur at other timescales. For example, consider the case of *loitering* - someone wanders at a place for a longer period. The investigation at a smaller time-step may not capture it, because at this timescale it appears as a normal walk. It can be captured only when observed for a sufficient amount of time. Hence, a larger timescale is needed to detect this long-term abnormal activity. Similarly, a larger timescale may not capture the short term anomaly (*e.g. jumping*) efficiently. In this paper, we propose a multi-timescale framework to address

This work was done when Royston Rodrigues and Neha Bhargava were at Indian Institute of Technology Bombay. The code and dataset will be made available at https://rodrigues-royston.github.io/Multi-timescale_Trajectory_Prediction

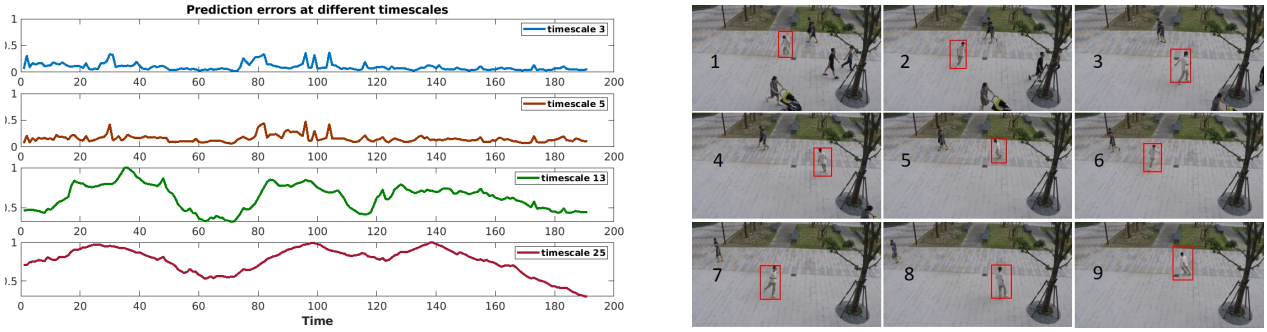


Figure 1. An illustration of how our model captures a long-term anomaly. The anomaly in consideration is *loitering* - the intermediate frames are shown on the right with the person involved in red box. The plots in the left show prediction errors at different timescales. The prediction errors are less at smaller timescales (3 and 5) because at these timescales, the model considers it as a normal activity (*walking*). The errors are higher at larger timescales (13 and 25). At these timescales, the model understands that it is an abnormal pattern of *walking*.

this problem. The framework has two models - one that makes future predictions and the other one that makes past predictions. They take in a single-person pose sequence to predict future and past sequences at multiple timescales. This is achieved by providing supervision at intermediate layers in the models. Since the model is trained at different timescales, it tends to learn temporal dynamics at various timescales. An illustration of a long-term anomaly (*e.g. loitering*) and prediction errors from our model at different timescales is shown in Figure 1.

Contributions: We make the following major contributions in the paper:

- We propose a bi-directional (past and future) prediction framework that considers an input pose trajectory and predicts pose trajectories at different timescales. Such a framework allows inspection of a trajectory at different timescales (*i.e.*, with different time duration).
- We introduce a large single-camera dataset that contains a diverse set of abnormal activities. Unlike other datasets, the dataset contains range of human anomalies - single person, multiple persons, or group.

The paper is organised as follows. The next section reviews the related work in the area of abnormal activity detection. Section 3 details the proposed model. We introduce our dataset in Section 4. We provide the experimental setup, ablation studies, and results in Section 5. Finally, the paper concludes in Section 6.

2. Related Work

The problem of human abnormal activity detection has been receiving a lot of interest from computer vision researchers. This is partly because of challenges inherent to the problem and mainly due to its applications. It is interesting to see the evolution of ideas over the years, especially after the introduction of deep learning in this area. In this section, we attempt to summarise this evolution with a few key papers. One of the common and initial approaches was based on reconstruction. Hasan *et al*. in [5] used an auto-encoder to learn appearance based representation under reconstruction framework. In [19], Xu *et al*. augmented the appearance features with optical flow to integrate motion information. Tran *et al*. learnt sparse representations by using convolutional winner-take-all autoencoders [17]. Luo *et al*. in [12] proposed a method based on learning a dictionary for the normal scenes under a sparse coding scheme. To smoothen the predictions over time, they proposed a Temporally-coherent Sparse Coding (TSC) formulation. Ravanbakhsh *et al*. used GAN to learn the normal representation [15]. In [9], Liu *et al*. used GAN with U-net to predict the future frames. Hinami *et al*. in [6] proposed a framework that jointly detects and recounts abnormal events. They also learn the basic visual concepts into the abnormal event detection framework. Abati *et al*. in [2] captured surprisal component of anomalous samples, by introducing an auto-regressive density estimator to learn latent vector distribution of autoencoders. In [18], Ionescu *et al*. used unmasking to detect anomalies without any training samples. Recently, Romera *et al*. in [13] has proposed a joint framework for trajectory prediction and reconstruction for anomaly detection. One major limitation of the existing methods is that they operate at a single timescale. We take a step forward and propose a multi-timescale model to address the limitations of operating with a single timescale.

3. Proposed Model

In this section, we present details of the proposed model. To restate, our objective is to develop a framework that is capable of detecting abnormal human activities of different time duration. Keeping this aim in mind, we propose a multi-timescale model that predicts the future trajectories at different timescales. The idea is to develop understanding at different timescales. To further improve the performance,

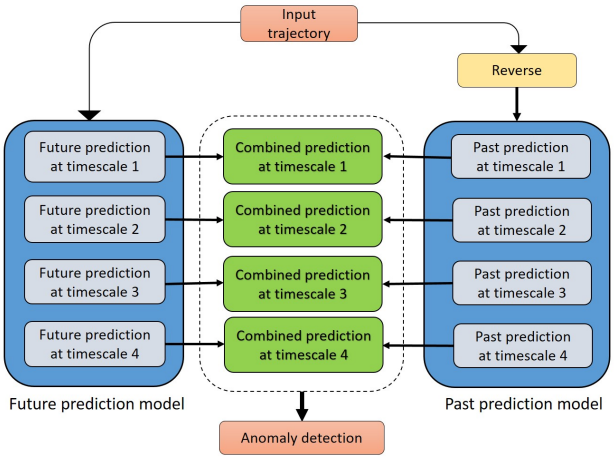


Figure 2. Top-level block diagram of the proposed framework. The future prediction model takes the input sequence and generate predictions at different timescales. To generate predictions at timescale 1, the model first splits the sequence into smaller sub-sequences and then makes future predictions for these sub-sequences. These predictions are combined to get the future prediction for the input sequence at this timescale. To get the past prediction, we reverse the input sequence and pass it to the past prediction model. Finally, all the predictions are combined appropriately to get a final prediction error for the input sequence that is used to detect abnormal events.

we add an identical model in the framework that reproduces the past. We use pose trajectory of human as the input. Besides being compact in nature, the pose trajectory captures the human movements quite well. A top-level block diagram of our framework is shown in Figure 2. It has two models that make past and future predictions, respectively. At a particular timescale, we combine the predictions from both the models to generate a combined prediction at every time instant. For example, to generate future predictions at timescale 1 (in our setup, timescale 1 represents time duration of 3 steps), the model first splits the sequence into smaller sub-sequences (of length 3) and then makes future predictions (for next 3 steps) for these sub-sequences. These predictions are combined to get the future prediction for the complete input sequence at this timescale. To get the past prediction, we reverse the input sequence and pass it to the past prediction model. Both models have the same architecture but are trained differently. The past and future predictions at a timescale are combined to get a predicted sequence at that timescale. Finally, all the predictions from different timescales are appropriately combined to get a final prediction error sequence for the input sequence. An illustration of our predicted poses is shown in Figure 3. It demonstrates higher prediction errors for abnormal activities and low errors for normal activities.

3.1. Problem Setup

We use human pose trajectory, represented by a collection of 25 points on the human skeleton over a time period, as the input to the model. Let $p^i_j(t) = [x^i_j(t), y^i_j(t)]$ be the image coordinates of j^{th} point in the pose representation of i^{th} person at time t. At time t, the pose of i^{th} person is represented as $X^i_t = [p^i_1(t), p^i_2(t), \ldots, p^i_{25}(t)]^T \in \mathbb{R}^{50\times 1}$ and the corresponding predicted pose is represented by $\hat{X}^i_t$. The model takes pose trajectory of a certain length as input and generates predictions at different timescales under a hierarchical framework. For any time instant at any timescale, the model generates multiple predictions because it runs a sliding window over the input signal. These multiple predictions are combined together by averaging to get a final prediction at a particular time instant. Similar approach is adopted to get the past predictions. Finally, the prediction errors from all the timescales are combined to get an abnormality score. In the next sub-section, we discuss the architecture.

3.2. Model

The proposed model is shown in Figure 4. The input to the model is the pose trajectory of an individual $\{X_t\}_{t=1,2,\ldots,T}$, where the superscript to identify a person is dropped for better clarity. The pose information at each time instant X_t is passed through an encoder $\mathbf{E}$ to get the corresponding encoded vector f_t of length 1024. The encoder $\mathbf{E}$ consists of two fully-connected layers of length 1024 to transform frame-level pose features of length 50 to 1024. The choice of 1024 is empirically decided. All these encoded vectors are passed to a series of 1D convolutional filters. We use 1D filters of length 3 for the initial two layers and then we have 1D filters of length 5 for the next five layers. We use 1024 filters at each layer. We train certain intermediate layers to produce predictions at different timescales. This is achieved by providing supervision at these layers. A specific layer corresponds to the k^{th} timescale if its reception field is of length t_k and is respon-

(a) *Walking* (b) *Cycling* (c) *Leaving bag* (d) *Throwing bag*

Figure 3. An illustration of predicted poses from the proposed model. The first scene has a *walking* action while the other three have abnormal actions. The *green* and *blue* poses represent the actual and predicted poses, respectively. The model generates large prediction errors for abnormal poses.

sible to generate t_k future (and past) steps in the trajectory. To provide supervision, we train a decoder to predict the future sequence. In our setup, we provide supervision to the layers corresponding to timescales - 3, 5, 13 and 25. The decoder $\mathbf{D_k}$ at time scale t_k consists of two fully-connected layers, of length 1024 and $t_k * 50$. The output length for the decoder depends on the specific timescale. Also, we use Relu as an activation function. Network architecture details are discussed in the supplementary material. In the next sub-section, we discuss the loss function in detail.

3.3. Loss Function

To compute the total loss of the model during training, we add the losses generated by the intermediate layers where we provide supervision. At such a layer, we have two types of losses - one at a node level, and another at the layer level. The loss at a node level computes the prediction error between the predictions generated by a node and the corresponding ground truth. The loss at a layer level computes the total prediction error generated by the layer for a complete input sequence. Since, there are multiple predictions generated at a particular time instant by different nodes, we use a sliding window approach to calculate the total loss at a layer. In particular, we take the average of all the prediction errors generated at a time instant by different nodes to compute the total prediction error at a particular time instant. The average prediction errors at all the time instants are added to get the layer loss. The total loss at the j^{th} layer with M_j number of nodes is given as,

$$\mathbb{L}_j = \sum_{i=1}^{M_j} L_1^j(i) + \sum_{t=1}^{T} L_2^j(t), \tag{1}$$

where $L_1^j(i)$ is the loss for i^{th} node and $L_2^j(t)$ is the loss at time t in the j^{th} layer. The first term corresponds to the total node loss and the second term corresponds to the total layer loss at j^{th} layer. Let the i^{th} node in j^{th} layer make predictions for the duration $\mathbb{T}(i) = [t_{si},\ t_{ei}]$ and generate prediction error $e(t, i)$ for a particular time instant $t \in [t_{si},\ t_{ei}]$. The i^{th} node loss is computed as follows:

$$L_1^j(i) = \sum_{t=t_{si}}^{t_{ei}} e(t, i) \tag{2}$$

To compute the layer loss, we simply take the average of prediction errors generated by different nodes for a particular time instant. We finally add these average errors for all the time instants to get the layer loss.

$$L_2^j = \frac{\sum_{i=1}^{M_j} e(t,i)\mathbb{1}(t \in \mathbb{T}(i))}{\sum_{i=1}^{M_j} \mathbb{1}(t \in \mathbb{T}(i))} \tag{3}$$

The total model loss for a model with N_{ts} number of timescales is,

$$Loss = \sum_{j=1}^{N_{ts}} \mathbb{L}_j \tag{4}$$

We use weighted mean square error (*mse*) to compute the prediction error.

$$e = \sum_{k=1}^{25} w_k(\hat{p}_k - p_k)^2, \tag{5}$$

where p_k, $\hat{p}_k$ are the original and predicted k^{th} pose points, respectively. The weight w_k is obtained from the confidence of pose estimator [4] for the k^{th} point as

$$w_k = \frac{c_k}{\sum_{i=1}^{25} c_i}, \tag{6}$$

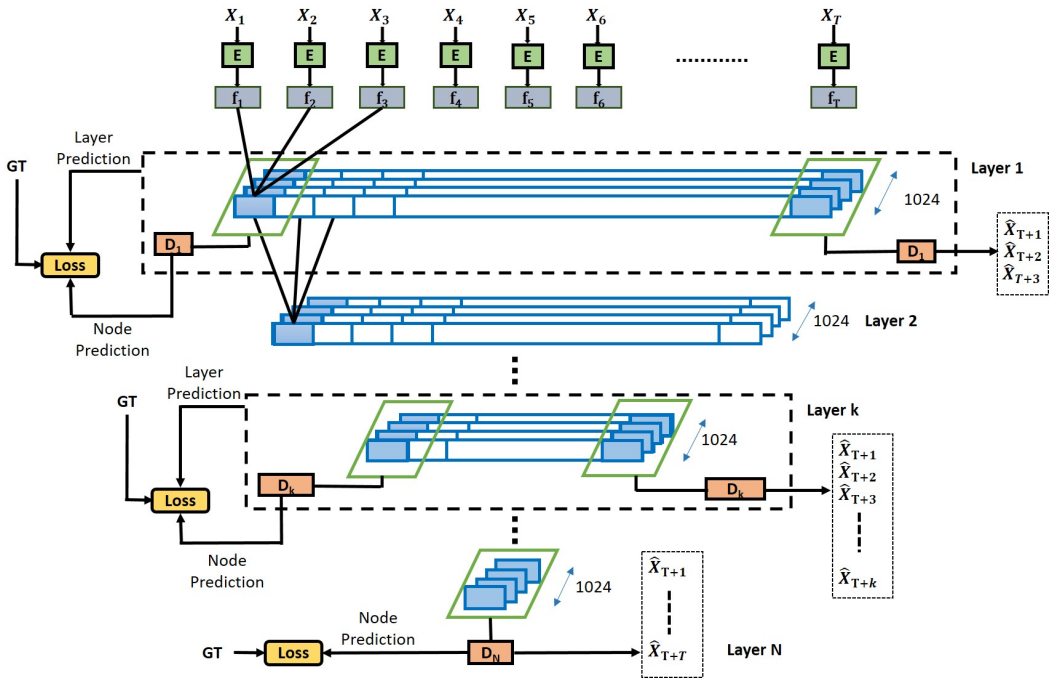


Figure 4. The detailed architecture of the proposed model for future prediction. X_1-X_T is the input pose trajectory. After encoding, the vectors f_1-f_T are passed to a series of 1D convolutional filters. A few intermediate layers generate predictions at different timescales. To generate predictions, the decoder takes the filtered encoded vector and produces the predictions $\hat{X}$. At these intermediate layers, the node loss and the layer loss are minimised jointly.

where c_k is the confidence score for the k^{th} point given by the pose detector.

With the loss function in Eq.(1), we jointly minimise the local loss at a node and the global loss at a layer. Minimising the node loss makes the prediction better at a node level, whereas minimising the layer loss drives the nodes to interact with each other to reduce the layer loss. Another advantage of using the layer loss under a sliding window approach during training is that it simulates the testing scenario. During testing, it is common to use a sliding window over a long test sequence to get an input sample of suitable length for the model.

3.4. Anomaly Detection

In this section, we discuss the method for anomaly detection during testing. The trained model predicts pose trajectories for a human at different timescales. So at any time instant, we have multiple predictions coming from different scales. These different prediction errors are combined using a voting mechanism to compute the final prediction error. At any time instant t, the errors are combined as follows:

$$error(t) = \frac{\sum_{j \in S} L_2^j(t)}{|S|}, \tag{7}$$

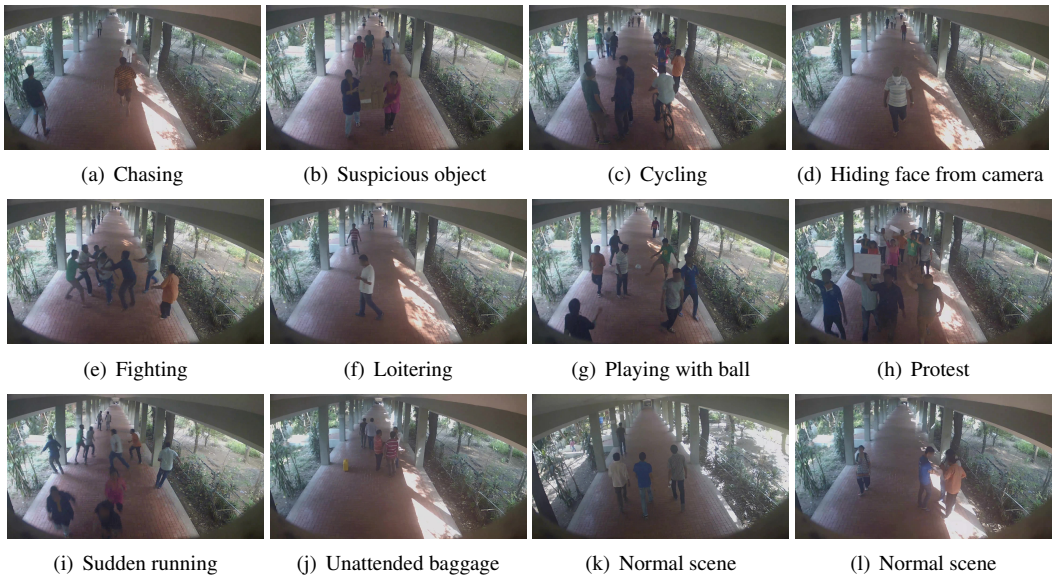
where S is a set of timescales that contains predictions for the time instant t and $L_2^j(t)$ is the j^{th} layer loss at time t, as in (3). Note that at any time instant, there can be more than one human resulting in multiple error plots - one for each human. In such a case, we take the maximum of prediction errors among all the individuals, at a time instant. That is,

$$error(t) = \frac{\sum_{j \in S} \max\{L_2^j(t, p_k), \forall k\}}{|S|}, \tag{8}$$

where $L_2^j(t, p_k)$ is the j^{th} layer loss at time t for the k^{th} individual. It is compared against a threshold to identify any abnormality.

4. IITB-Corridor Dataset

We provide a brief introduction of the proposed *IITB-Corridor* dataset for abnormal human activity. The videos are captured in IIT Bombay campus under a single-camera setup. The scene consists of a corridor where the normal activities are usually *walking* and *standing*. We enacted various abnormal activities with the help of volunteers. The dataset contains variety of activities and has single person to group level anomalies. The annotations for normal and abnormal are provided at frame level. A comparison of the proposed dataset with other datasets is given in Table 1. The last column mentions the abnormal activities present in the datasets. Out of 1,81,567 test frames, the number of abnormal frames is 1,08,278. Another large dataset is proposed



(a) Chasing (b) Suspicious object (c) Cycling (d) Hiding face from camera

(e) Fighting (f) Loitering (g) Playing with ball (h) Protest

(i) Sudden running (j) Unattended baggage (k) Normal scene (l) Normal scene

Figure 5. Sample images from the proposed *IITB-Corridor* dataset.

Camera Specific Datasets			
Dataset	**Training frames**	**Testing frames**	**Abnormal activities**
USCD Ped-1 [8]	6,800	7,200	Bikers, small carts, walking across walkways
USCD Ped-2 [8]	2,550	2,010	Bikers, small carts, walking across walkways
Subway [3]	20,000	116,524	Climbing over fence, wrong direction
Avenue [11]	15,328	15,324	Running, Throwing object, Wrong direction
ShanghaiTech [12]	2,74,515	42,883	Throwing Object, Jumping, Pushing Riding a Bike, Loitering, Climbing
IITB-Corridor [1] (proposed)	3,01,999	1,81,567	Protest, Unattended Baggage, Cycling, Sudden Running, Fighting, Chasing, Loitering, Suspicious Object,Hiding, Playing with Ball
Camera Independent Datasets			
UCF Anomaly Detection [16]	1610 Training videos	290 Testing videos	Abuse, Arrest, Arson, Assault, Accident, Burglary, Explosion, Fighting, Robbery, Shooting, Stealing, Shoplifting, and Vandalism

Table 1. *IITB-Corridor* (proposed) dataset compared to existing single camera datasets. This dataset is more challenging as can be seen in the Table 2 showing performance of state-of-the-art methods on different datasets.

by [16] that was mined from YouTube and LiveLeak but it is not camera specific. We also provide class label for each abnormal activity that is useful for classification purposes. Figure 5 shows sample images from the dataset corresponding to all the activities. To keep the privacy of volunteers intact, we have blurred the faces in the images. We used the algorithm proposed by [14] to detect the faces. In the next section, we discuss our experimentation in detail.

5. Results and Discussions

In this section, we discuss our experimental setup and results. We tested our method on the proposed dataset and two public datasets namely, ShanghaiTech Campus dataset [12] and Avenue [11]. The ShanghaiTech Campus dataset contains videos from 13 different cameras around the ShanghaiTech University campus. A few examples of human anomalies present in the dataset are *running*, *fighting*, *loitering*, and *jumping*. Avenue dataset is captured at CUHK campus. It contains anomalies such as *throwing a bag*, *running*, *walking near the camera*, and *dancing*. The training set has 16 videos and the test set has 21 videos. Since we are interested in human anomaly, similar to [13], we test our algorithm primarily on HR-ShanghaiTech and HR-Avenue datasets, proposed by them. In this section, we first discuss our pre-processing to generate the pose trajectories, followed by training and testing schemes. We then compare the performance of our model with the state-of-the-art methods.

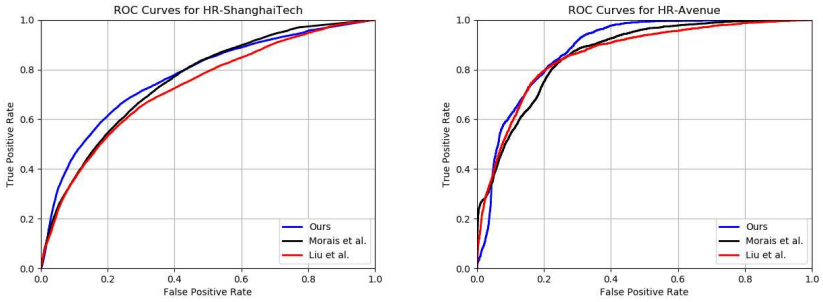


Figure 6. ROC plots for HR-ShanghaiTech and HR-Avenue dataset obtained from Morais *et al*. [13], Liu *et al*. [9], and our method.

	HR-ShanghaiTech	ShanghaiTech	HR-Avenue	Avenue	*IITB-Corridor*
Conv-AE [5]	69.80	70.40	84.80	70.20	-
Ionescu *et al*. [18]	-	-	-	80.60	-
TSC-rRNN [12]	-	68.00	-	81.71	-
Liu *et al*. [9]	72.70	72.80	86.20	**84.90**	64.65
Abati *et al*. [2]	-	72.50	-	-	-
Morais *et al*. [13]	75.40	73.40	86.30	-	64.27
Proposed	**77.04**	**76.03**	**88.33**	82.85	**67.12**

Table 2. Performance comparison with other existing techniques. In most datasets, the proposed method outperforms other methods.

5.1. Data Preparation

In this section, we discuss our method to obtain the pose trajectories from the videos. We first run a human detector [7] on all the videos. We use the multi-target tracker proposed by [10] to obtain bounding box trajectories from the detections. Finally, we run a pose-detector [4] on these bounding boxes to get the pose trajectories. This pose detector outputs the locations of 25 points on human bounding box. It also produces confidence values for each of 25 points. We use these values in Eq. (5) to calculate the weighted *mse*.

5.2. Training

In this section, we discuss our training paradigm. To generate predictions at different timescales, we provide supervision at pre-chosen layers in our multi-layered model. In our model, we provide supervision to the layers - 1, 2, 4, and 7 corresponding to timescales of 3, 5, 13, and 25, respectively. Each training epoch consists of sub-epochs. In each sub-epoch, we train up to a particular layer corresponding to one of the timescales. In the first sub-epoch, we train only the first layer corresponding to timescale of 3. In the last sub-epoch, we train the complete model up to layer 7 that corresponds to timescale of 25. There are 4 sub-epochs corresponding to each timescale. To train the first sub-epoch, we split the training pose trajectories in to smaller trajectories of length 6 (3 in and 3 out). To train the last sub-epoch, we split the input trajectories in to length of 50 (25 in and 25 out). The loss after an epoch is equal to the loss incurred at the last sub-epoch. We used *Adam* as the optimiser to train the model.

5.3. Testing

To test an input sequence, we split the input sequence in to smaller sequences of length 6, 10, 26, and 50. We use sequences of length 6 to generate predictions from layer 1 (*i.e*., timescale of 3). Similarly, we use sequences of lengths 10, 26, and 50 to produce predictions at layer 2, 4, and 7, respectively. This is done for both future and past prediction models. We combine the prediction errors by voting. The forward pass to generate predictions at all the scales from one model takes 50μs on NVIDIA RTX 2070. To generate all the prediction errors, it needs to wait for next 25 frames. That is, it takes around 1s (for 25 *fps*) to make inference at any time instant.

5.4. Performance Evaluation

We compare our results with [12], [9], [5], [18], [2], and [13]. The method proposed by Luo *et al*. in [12] is based on learning a dictionary for the normal scenes under a sparse coding scheme. To smoothen the predictions over time, they proposed a Temporally-coherent Sparse Coding (TSC) formulation. Liu *et al*. [9] proposed a future frame prediction based method to detect anomaly. The method proposed by Morais *et al*. [13] uses pose trajectory under the joint framework of reconstruction and prediction. To compare with these existing approaches, we use *Frame-AUC* as the evaluating criteria. The comparison is given in Table 2. The proposed model outperforms the other methods on HR-ShanghaiTech and HR-Avenue datasets.

Timescales	HR-ShanghaiTech			HR-Avenue		
	Future	Past	Future+Past	Future	Past	Future+Past
3	71.71	70.62	72.05	85.33	83.36	84.99
3, 5	72.89	71.69	73.39	86.96	84.70	86.82
3, 5, 13	74.51	73.39	75.65	**88.29**	**86.20**	**88.43**
3, 5, 13, 25	**74.98**	**74.17**	**77.04**	87.37	85.65	88.33

Table 3. Effect of multiple timescales and past predictions on the overall performance of the model. It can be seen that both are needed for improved performance.

Loss	HR-ShanghaiTech			HR-Avenue		
	Future	Past	Future+Past	Future	Past	Future+Past
Layer loss	73.66	74.03	76.46	87.09	85.15	88.05
Node loss	74.26	74.03	76.67	87.31	85.19	88.12
Layer and Node losses	**74.98**	**74.17**	**77.04**	**87.37**	**85.65**	**88.33**

Table 4. Effect of layer loss and node loss on overall performance of the model. It is observed that using both the losses improved the performance.

Even though our model doesn't capture the non-human anomalies (*e.g*. car), it outperforms on ShanghaiTech dataset and provides comparable performance on Avenue dataset. The ROC plots are provided in Figure 6.

5.5. Ablation Studies

In this section, we provide ablation studies. In particular, we highlight the importance of multiple timescales, past prediction model, and the choice of loss function.

5.5.1 Effect of Multi-timescale Framework

In order to understand the importance of multiple timescales, we use our trained model and then test the model with incremental combinations of timescales. Table 3 compares the *Frame AUC* when using different timescales. In most of the cases, we see an improvement in performance as we include more timescales. However, in case of HR-Avenue, we do not see this trend in the last row as this dataset does not have long-term anomalies corresponding to timescale of 25.

5.5.2 Effect of Past Prediction Model

To see the effect of past prediction model, we compare the performances of future prediction model, past prediction model, and combined one. In Table 3, we can observe from the rows that adding past prediction model improved the overall performance in most cases.

5.5.3 Effect of Layer and Node Losses

To see the effect of adding layer loss to the node loss, we compare performance of the model trained using only the layer loss, node loss, and the combined. In Table 4, we observed that there is a slight improvement when both the losses are used. We used the complete model with 4 timescales (3, 5, 13, and 25) for this ablation study.

6. Conclusions

In this work, we developed a multi-timescale framework to capture abnormal human activities occurring at different timescales. The multi-timescale predictions are used to detect abnormal human activity. Our experiments show that the proposed framework outperforms state-of-the-art models. In addition, we also release a challenging single-camera abnormal activity data set that has varieties of abnormalities. In this paper, we addressed single person based anomaly and our future work will focus on anomalies involving multiple persons.

7. Acknowledgment

The authors are thankful for the efforts by Ashvini Sharma and Yashswi Jain towards preparing the IITB Corridor dataset. This research work was supported by National Center of Excellence in Technology for Internal Security (NCETIS) an initiative by IIT Bombay and Ministry for Electronics and Information Technology (MeitY).

References

[1] `https://rodrigues-royston.github.io/Multi-timescale_Trajectory_Prediction`. Online: accessed 13/01/2020.

[2] D. Abati, A. Porrello, S. Calderara, and R. Cucchiara. Latent space autoregression for novelty detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2019.

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

[4] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2D pose estimation using part affinity fields. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[5] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis. Learning temporal regularity in video sequences. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 733–742, 2016.

[6] R. Hinami, T. Mei, and S. Satoh. Joint detection and recounting of abnormal events by learning deep generic knowledge. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3619–3627, 2017.

[7] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7310–7311, 2017.

[8] W. Li, V. Mahadevan, and N. Vasconcelos. Anomaly detection and localization in crowded scenes. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 36(1):18–32, 2013.

[9] W. Liu, W. Luo, D. Lian, and S. Gao. Future frame prediction for anomaly detection–a new baseline. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6536–6545, 2018.

[10] C. Long, A. Haizhou, Z. Zijie, and S. Chong. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *IEEE International Conference on Multimedia and Expo*, 2018.

[11] C. Lu, J. Shi, and J. Jia. Abnormal event detection at 150 fps in Matlab. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2720–2727, 2013.

[12] W. Luo, W. Liu, and S. Gao. A revisit of sparse coding based anomaly detection in stacked RNN framework. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 341–349, 2017.

[13] R. Morais, V. Le, T. Tran, B. Saha, M. Mansour, and S. Venkatesh. Learning regularity in skeleton trajectories for anomaly detection in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11996–12004, 2019.

[14] M. Najibi, P. Samangouei, R. Chellappa, and L. Davis. SSH: Single stage headless face detector. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[15] M. Ravanbakhsh, M. Nabi, E. Sangineto, L. Marcenaro, C. Regazzoni, and N. Sebe. Abnormal event detection in videos using generative adversarial nets. In *IEEE International Conference on Image Processing*, pages 1577–1581. IEEE, 2017.

[16] W. Sultani, C. Chen, and M. Shah. Real-world anomaly detection in surveillance videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6479–6488, 2018.

[17] H. T. Tran and D. Hogg. Anomaly detection using a convolutional winner-take-all autoencoder. In *Proceedings of the British Machine Vision Conference 2017*, 2017.

[18] R. Tudor Ionescu, S. Smeureanu, B. Alexe, and M. Popescu. Unmasking the abnormal events in video. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2895–2903, 2017.

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