FastAno: Fast Anomaly Detection via Spatio-temporal Patch Transformation

Chaewon Park  MyeongAh Cho  Minhyeok Lee  Sangyoun Lee *
Yonsei University, Seoul, Republic of Korea
{chaewon28, maycho0305, hydragon516, syleee}@yonsei.ac.kr

Abstract

Video anomaly detection has gained significant attention due to the increasing requirements of automatic monitoring for surveillance videos. Especially, the prediction based approach is one of the most studied methods to detect anomalies by predicting frames that include abnormal events in the test set after learning with the normal frames of the training set. However, a lot of prediction networks are computationally expensive owing to the use of pre-trained optical flow networks, or fail to detect abnormal situations because of their strong generative ability to predict even the anomalies. To address these shortcomings, we propose spatial rotation transformation (SRT) and temporal mixing transformation (TMT) to generate irregular patch cuboids within normal frame cuboids in order to enhance the learning of normal features. Additionally, the proposed patch transformation is used only during the training phase, allowing our model to detect abnormal frames at fast speed during inference. Our model is evaluated on three anomaly detection benchmarks, achieving competitive accuracy and surpassing all the previous works in terms of speed.

1. Introduction

Video anomaly detection refers to the task of recognizing unusual events in videos. It has gained attention due to the implementation of video surveillance systems. Surveillance cameras are widely used for public safety. However, the monitoring capacity is not up to the mark. Since abnormal events rarely happen in the real world compared to normal events, automatic anomaly detection systems are in high demand to reduce the monitoring burden. However, it is very challenging because obtaining the datasets is difficult owing to the imbalance of events and variable definitions of abnormal events based on the context of each video.

*Corresponding Author.
This work was supported by the Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2021-0-00172, The development of human Re-identification and masked face recognition based on CCTV camera)
capacity of convolutional neural networks (CNNs) [12]. To minimize this factor, Gong et al. [12] and Park et al. [34] proposed memory-based methods to use only the most essential features of normal frames for the generation. However, the memory-based methods are not efficient for videos with various scenes because their performance is highly dependent on the number of items. Many memory items are required to read and update patterns of various scenes, slowing down the detection.

Another critical and challenging issue for video anomaly detection is to detect abnormal events or emergencies immediately, but slow models do not meet this purpose. In the previous studies, the following factors are observed to slow down the detection speed: heavy pre-trained networks such as optical flow [22, 39, 40, 49], object detectors [10, 15], and pre-trained feature extractors [38, 42]. These modules are complex and computationally expensive. Therefore, we take the detection speed into account and employ a patch transformation method that is used only during training. We implement this approach by artificially generating abnormal patches via applying transformations to patches randomly selected from the training dataset. We adopt spatial rotation transformation (SRT) and temporal mixing transformation (TMT) to generate a patch anomaly at a random location within a stacked frame cuboid. Given this anomaly-included frame cuboid, our AE is trained to learn the employed transformation and predict the upcoming normal frame. The purpose of SRT is to generate an abnormal appearance and encourage the model to learn spatially invariant features of normal events. For instance, when a dataset defines walking pedestrians as normal and all others as abnormal, by giving a sequence of a rotated person (e.g., upside-down, lying flat) and forcing the model to generate a normally standing person, the model learns normal patterns of pedestrians. TMT, which is shuffling the selected patch cube in the temporal axis to create abnormal motion, is intended to enhance learning temporally invariant features of normal events. Given a set of frames where an irregular motion takes place in a small area, the model has to learn how to rearrange the shuffled sequence in the right order to correctly predict the upcoming frame.

To the best of our knowledge, unlike [22, 12, 34, 39, 15, 38], our framework performs the fastest because there are no additional modules or pre-trained networks. Furthermore, the proposed patch transformation does not drop the speed because it is detached during detection. Likewise, we designed all components of our method considering the detection speed in an effort to make it suitable for anomaly detection in the real world.

We summarize our contributions as follows:

- We apply a patch anomaly generation phase to the training data to enforce normal pattern learning, especially in terms of appearance and motion.
- The proposed patch generation approach can be implemented in conjunction with any backbone network during the training phase.
- Our model performs at very high speed and at the same time achieves competitive performance on three benchmark datasets without any pre-trained modules (e.g., optical flow networks, object detectors, and pre-trained feature extractors).

2. Related work

2.1. AE-based approach

Frame predicting and reconstructing AEs have been proposed under the assumption that models trained only on normal data are not capable of predicting or reconstructing abnormal frames, because these are unseen during training. Some studies [22, 43, 34, 26] trained AEs that predict a single future frame from several successive input frames. Additionally, many effective reconstructing AEs [33, 51, 34, 5] have been proposed. Cho et al. [5] proposed two-path AE, where two encoders were used to model appearance and motion features. Focusing on the fact that abnormal events occur in small regions, patch-based AEs [51, 47, 33, 8], have been proposed. However, it has been observed that AEs tend to generalize well to generate abnormal events strongly, mainly due to the capacity of CNNs, which leads to missing out on anomalies during detection. To alleviate this drawback, Gong et al. [12] and Park et al. [34] suggested networks that employ memory modules to read and update memory items. These methods showcased outstanding performance on several benchmarks. However, they are observed to be ineffective for large datasets due to the limitation of memory size. Furthermore, some works [22, 39, 40, 38] have used optical flow to estimate motion features because information of temporal patterns is crucial in anomaly detection.

2.2. Transformation-based approach

Many image transformation methods, such as augmentations, have been proposed to increase recognition performance and robustness in varying environments in limited training datasets. This technique was first applied to image recognition and was later extended to video recognition. For the image-level modeling, Komodakis et al. [19] suggested unsupervised learning for image classification by predicting the direction of the rotated input. Krizhevsky et al. [20] used rotation, flipping, cropping, and color jittering to enhance learning spatially invariant features. Furthermore, DeVries et al. [7] devised CutOut, a method that deletes a box at a random location to prevent the model from focusing only on the most discriminative regions. Zhang et
al. [52] proposed MixUp which blends two training data on both the images and the labels. Yun et al. [50] put forth a combination of CutOut and MixUp, called CutMix. For the video-level model, augmentation techniques have been extended to the temporal axis. Ji et al. [16] proposed a method called time warping and time masking, which randomly skips or adjusts temporal frames.

Several studies have used the techniques mentioned above for video anomaly detection based on the assumption that applying transformations to the input forces the network to embed critical information better. Zaheer et al. [51] suggested a pseudo anomaly module to create an artificial anomaly patch by blending two arbitrary patches from normal frames. They reconstructed both normal and abnormal patches and trained a discriminator to predict the source of the reconstructed output. Hasan et al. [13] and Zhao et al. [54] sampled the training data by skipping a fixed number of frames in the temporal axis. Moreover, Joshi et al. [17] generated abnormal frames from normal frames by cropping an object detected with a semantic segmentation model and placing it in another region in the frame to generate an abnormal appearance. Wang et al. [46] applied random cropping, flipping, color distortion, rotation, and grayscale to the entire frame. In contrast to these methods, our network embeds normal patterns by training from frames with anomaly-like patches. We transform the input frames along the spatial axis or temporal axis to generate abnormal frames within training datasets. Georgescu et al. [10] used sequence-reversed frames for a self-supervised binary classification task where the network guesses whether the given samples are regular or not. On the other hand, our method predicts the original frame from sequence transformed input and learns the normal patterns.

3. Proposed approach

This section presents an explicit description of our model formation. Our model consists of two main phases: (1) the patch anomaly generation phase and (2) the prediction phase.

3.1. Overall architecture

Fig. 2 presents the overview of our framework. During the training phase, we first load \( n \) adjacent frames to make a frame cuboid. After that, we apply our patch anomaly generation to the frame cuboid, which is forwarded to the AE. Our AE extracts spatial and temporal patterns of the input and generates a future frame. During inference, the patch anomaly generation is not employed. A raw frame cuboid is fed as an input to the AE. The difference between the output of the AE and the ground truth frame is used as a score to judge normality.

3.2. Patch anomaly generation phase

Abnormal events in videos are categorized into two large branches: (1) anomalies regarding appearances (e.g., pedestrians on a vehicle road) and (2) anomalies regarding motion (e.g., an illegal U-turn or fighting in public). Hence, it is important to learn both the appearance and motion features of normal situations to detect anomalies in both cases.
The patch anomaly generation phase takes place before feeding the frames to the generator. We load \( n \) successive frames \((F_t, F_{t+1}, F_{t+2}, \ldots, F_{t+n-1})\), resize each to \( 240 \times 360 \), and concatenate them on the temporal axis to form a 4D cuboid \( C_F \in \mathbb{R}^{C \times n \times 240 \times 360} \), where \( C \) denotes the number of channels for each frame. After that, we select a patch cuboid \( C_p \in \mathbb{R}^{C \times \delta \times \delta} \) from a random location within \( C_F \) to apply transformation. Since anomalies usually occur in foregrounds, we exclude a margin of 12.5 percent in length from the top and bottom of the width of \( C_F \) from the selection area. We heuristically find that these marginal regions are generally backgrounds. Therefore, they commonly do not contain moving objects. Thus, by limiting the range, \( C_p \) is more likely to capture foregrounds than backgrounds, encouraging the model to concentrate on the moving objects. Then we apply SRT or TMT to \( C_F \) to form a transformed patch cuboid \( C'_p \). Only one of the two is applied randomly for every input.

For SRT, each patch is rotated in a random direction between \( 0^\circ, 90^\circ, 180^\circ, \) and \( 270^\circ \), following the approach of [19]. By forwarding these transformed frame cuboids \( C_F \) \((F'_t, F'_{t+1}, F'_{t+2}, \ldots, F'_{t+n-1})\) to the frame generator, our network is encouraged to focus on the abnormal region and recognize the spatial features of the normal appearances. Suppose a network is being trained on a dataset of people walking on a road. When it is given a frame cuboid with \( n \) normal pedestrians and is programmed to predict a walking person is generated within a frame where only for- ward walking people are annotated as normal, the model needs to detect the awkward motion and match the sequence to normal before predicting the next frame to reduce the loss and generate a frame as similar as possible to the ground truth. For example, when the patch sequence is reversed, and a backward-walking person is generated within a frame where only forward walking people are annotated as normal, the model should find the correct sequence of the abnormal person based on the learned features to predict the correct trajectory.

Our TMT function is as follows:

\[
TMT(F_i) = T(F_i; F'_{i\delta}^\xi) \in \mathbb{R}^{C \times \delta \times \delta}, \quad (2)
\]

where \( T \) denotes a function that copies a patch located in pixel range of \([x, x + W_p]\) in the width axis and \([y, y + H_p]\) in the height axis of input frame \( F_{i\delta} \) and pastes it to the \( i^{th} \) frame. \( \xi \) represents the shuffled sequence of \( n \) patches (e.g., sequence \((4, 1, 0, 3, 2)\) when \( n \) is 5). Same as SRT, the final \( C'_p \) is the stack of the transformed \( F_i \).

Our patch anomaly generation phase is computationally cheaper than the other methods that embed spatio-temporal feature extraction in networks, such as storing and updating memory items \([12, 34]\), and estimating optical flow with pre-trained networks \([22, 39, 49, 2]\). Therefore, our patch anomaly generation phase boosts feature learning at a low cost. Furthermore, this phase is not used during the inference, meaning that it does not affect the detection speed at all. Thus, our model is low in complexity and computational costs (see Section 4.3).

3.3. AE architecture

The AE in our network aims to learn prototypical features of normal events and produce an output frame based on those features. Its main task is to predict \( P_t \)—the frame coming after \( C_F \)—from an input frame cuboid \( C'_F \). Therefore, it is necessary to learn the temporal features as well as the spatial features to generate the frame with fine quality. The architecture of our model follows that of U-Net [41], in which the skip connections between the encoder and the decoder boost generation ability by preventing gradient vanishing and achieving information symmetry. The encoder consists of a stack of three-layer blocks that reduce the resolution of the feature map. We employ 3D convolution [44] to embed the temporal factor learning in our model. Specific-
ically, the first block consists of one convolutional layer and one activation layer: convolutional, batch normalization, and activation layers. The kernel size is set to $3 \times 3 \times 3$ for all three layers. The decoder also consists of a stack of three-layer blocks and is symmetrical to the encoder except that the convolutional layers are replaced by deconvolutional layers to upscale the feature map. In addition, we use leakyReLU activation [29] for the encoder and ReLU activation [31] for the decoder.

Likewise, the architecture of our AE is very simple compared to other previous studies, especially methods that employ pre-trained feature extractors [28, 42]. In that the running time is generally dependent on the simplicity of the model architecture, our AE is well designed, considering the speed.

3.4. Objective function and normality score

Prediction loss. Our model is trained to minimize the prediction loss. We use the $L1$ distance (Eq. (3)) and structural similarity index (SSIM) [45] loss (Eq. (4)) to measure the difference between the generated frame $P'_t$ and the ground truth frame $P_t$. The $L1$ distance and SSIM demonstrate the difference of frames at the pixel-level and similarity at the feature-level, respectively. The functions are as follows:

$$L_p(P'_t, P_t) = |P'_t - P_t|$$  \hspace{1cm} (3)

$$L_f(P'_t, P_t) = 1 - \frac{(2\mu_{P'_t}\mu_{P_t} + c_1)(2\sigma_{P'_t}\sigma_{P_t} + c_2)}{(2\mu_{P'_t}^2 + c_1)(2\sigma_{P'_t}^2 + c_2)}$$  \hspace{1cm} (4)

where $\mu$ and $\sigma^2$ denote the average and variance of each frame, respectively. Furthermore, $\sigma_{P'_t}P_t$ represents the co-variance. $c_1$ and $c_2$ denote variables to stabilize the division. Following the work of Zhao et al. [53], we exploit a weighted combination of the two loss functions in our objective function as shown in Eq. (5).

$$L_{pred}(P'_t, P_t) = \omega_p L_p(P'_t, P_t) + \omega_f L_f(P'_t, P_t)$$  \hspace{1cm} (5)

$\omega_p$ and $\omega_f$ are the weights controlling the contribution of $L_p$ and $L_f$, respectively. Consequently, our model is urged to generate outputs that resemble the ground truth frames at both the pixel and feature levels.

Frame-level anomaly detection. When detecting anomalies in the testing phase, we adopt the peak signal to noise ratio (PSNR) as a score to estimate the abnormality of the evaluation set. We obtain this value between the predicted frame at the $t^{th}$ period $P'_t$ and the ground truth frame $P_t$:

$$PSNR(P'_t, P_t) = 10 \log_{10} \frac{\text{max}(P'_t)}{\|P'_t - P_t\|_2^2/N}$$  \hspace{1cm} (6)

where $N$ denotes the number of pixels in the frame. Our model fails to generate when $P_t$ contains abnormal events, resulting in a low value of PSNR and vice versa. Following the method of many related studies [6, 12, 13, 21, 22, 28, 34, 40], we define the final normality score $S_t$ by normalizing $PSNR(P'_t, P_t)$ of each video clip to the range $[0, 1]$.

$$S_t = \frac{PSNR(P'_t, P_t) - \min PSNR(P'_t, P_t)}{\max PSNR(P'_t, P_t) - \min PSNR(P'_t, P_t)},$$  \hspace{1cm} (7)

Therefore, our model is capable of discriminating between normal and abnormal frames using the normality score of Eq. (7).

4. Experiments

4.1. Implementation details

We implement all of our experiments with PyTorch [35], using a single Nvidia GeForce RTX 3090. Our model is trained using Adam optimizer [18] with a learning rate of 0.0002. Additionally, a cosine annealing scheduler [23] is used to reduce the learning rate to 0.0001. We train our model for 20 epochs on the Avenue dataset [24] and Ped2 dataset [30] and five epochs on the ShanghaiTech dataset [28]. The number of input frames $n$ is empirically set to 5. We load frames in gray scale in order to improve the speed and efficiency. Then we resize the frames to $240 \times 360$, and normalize the intensity of pixels to $[-1, 1]$. In addition, we add random Gaussian noise to the training input where the mean is set to 0 and the standard deviation is chosen randomly between 0 and 0.03. Furthermore, we set $W_p$ and $H_p$ to 60. The batch size is 4 during training. Optimal weights for the loss function in Eq. (5) are empirically measured as $\omega_p = 0.25$ and $\omega_f = 0.75$.

Evaluation metric. We adopt the area under curve (AUC) of the receiver operating characteristic (ROC) curve obtained from the frame-level scores and the ground truth labels for the evaluation metric. This metric is used in most studies [4, 10, 11, 27, 28, 33, 34, 49, 51] on video anomaly detection. Some works also report the localizing performance by adopting the pixel-level AUC. However, according to Ramachandra et al. [37], this criterion is a flawed metric because the results can be artificially improved by using some expedient tricks. Furthermore, this metric does not penalize the false positive detection within the true positive frames, meaning that the corresponding results are actually unsuitable for representing the spatial detecting performance. Therefore, new metrics called the Track-Based Detection Rate (TBDR) and the Region-Based Detection Rate (RBDR) [36] were proposed recently to replace the pixel-level AUC. However, the official implementation data have yet to be released. Hence, we only consider the temporal evaluation in this paper.

The baseline model, mentioned throughout the following sections, denotes our model without the patch anomaly
4.2. Datasets

We evaluate our model with three datasets which are all acquired from the real-world scenarios.

**CUHK Avenue [24].** This dataset captures an avenue at a campus. It consists of 16 training and 21 testing clips. Training clips contain only normal events and testing clips contain a total of 47 abnormal events such as running, loitering, and throwing objects. The frame resolution is 360 × 640, all in RGB scale. The size of people is inconsistent due to the camera angle. Furthermore, the camera is kept fixed most of the time. However, a subtle shaking is recorded briefly in the test set.

**UCSD Ped2 [30].** The UCSD Ped2 dataset [30] is acquired from a pedestrian walkway by a fixed camera from a long distance. The training and the testing sets consist of 16 and 12 clips, respectively. Anomalies in the testing clips are non-pedestrian objects, for instance, bikes, cars, and skateboards. The frames are in gray scale with a resolution of 240 × 360.

**ShanghaiTech Campus [28].** Unlike the others, this dataset contains multi-scene anomalies and is the most complex and largest dataset. It is acquired from 13 different scenes. There are 330 training videos and 107 testing videos where non-pedestrian objects (e.g., cars, bikes) and aggressive motions (e.g., brawling, chasing) are annotated as anomalies. Each frame is captured with 480 × 856 RGB pixels.

4.3. Experimental results

**Impact of patch anomaly generation phase.** Table 2 shows the impact of our patch anomaly generation estimated on Avenue [24] and Ped2 [30]. The results include five different conditions: (1) using only TMT, (2) using only SRT, (3) randomly applying TMT or SRT but with all patches rotated as a chunk in the same direction for SRT, where \( \delta_t = \delta_{t+1} = \cdots = \delta_{n-1} \) (represented as SRT* in Table 2), (4) randomly applying TMT or SRT with varying directions for each patch, and (5) applying both TMT and SRT to the selected \( \mathbf{C}_P \). From the results, it appears that SRT has a greater contribution than TMT to the detection performance. This is because our SRT rotates each patch randomly in varying directions resulting in generating anomalies in the motion as well as the appearance.

**Performance comparison with existing works.** We compare the frame-level AUC of our model with those of non-prediction-based methods [13, 28, 39, 42, 40, 33, 12, 15, 34, 49, 51, 10, 11] and prediction-based methods [22, 43, 32, 34]. From Table 1, we find that our method achieves

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>StackRNN [28]</td>
<td>50</td>
<td>✓</td>
<td>81.7</td>
<td>68.0</td>
<td>92.2</td>
</tr>
<tr>
<td>FFP [22]</td>
<td>133</td>
<td>✓</td>
<td>85.1</td>
<td>72.8</td>
<td>95.4</td>
</tr>
<tr>
<td>AD [40]</td>
<td>2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>95.5</td>
</tr>
<tr>
<td>AMc [32]</td>
<td>✓</td>
<td>86.9</td>
<td>88.5</td>
<td>32, 34</td>
<td></td>
</tr>
<tr>
<td>MemAE [12]</td>
<td>42</td>
<td>✓</td>
<td>83.3</td>
<td>71.2</td>
<td>94.1</td>
</tr>
<tr>
<td>DummyAno [15]</td>
<td>11</td>
<td>✓</td>
<td>87.4</td>
<td>78.7†</td>
<td>94.3†</td>
</tr>
<tr>
<td>AnoPCN [48]</td>
<td>10</td>
<td>✓</td>
<td>86.2</td>
<td>73.6</td>
<td>96.8</td>
</tr>
<tr>
<td>GLNC [55]</td>
<td>150</td>
<td>-</td>
<td>84.1</td>
<td>-</td>
<td>92.8</td>
</tr>
<tr>
<td>GMV AE [8]</td>
<td>120</td>
<td></td>
<td>83.4</td>
<td>-</td>
<td>92.2</td>
</tr>
<tr>
<td>VECVAD [49]</td>
<td>2</td>
<td></td>
<td>90.6</td>
<td>74.8</td>
<td>97.3</td>
</tr>
<tr>
<td>FewShotGAN [27]</td>
<td>✓</td>
<td>85.8</td>
<td>77.9</td>
<td>96.2</td>
<td></td>
</tr>
<tr>
<td>AMmem [2]</td>
<td></td>
<td>86.6</td>
<td>73.7</td>
<td>96.6</td>
<td></td>
</tr>
<tr>
<td>MTL [10]</td>
<td>21</td>
<td>✓</td>
<td>91.5†</td>
<td>82.4†</td>
<td>97.5†</td>
</tr>
<tr>
<td>BackAgnostic [11]</td>
<td>18</td>
<td></td>
<td>92.3</td>
<td>82.7</td>
<td>98.7</td>
</tr>
</tbody>
</table>

Table 1. Frame-level AUC scores (%) of the state-of-the-art methods versus our architecture trained with patch anomaly generation phase. For a fair comparison, like all other papers, the * marked scores are the micro-AUC performances taken from [9, 11]. The FPS values are based on the figures mentioned in each paper, and the ones with † denote FPS computed in our re-implementation, conducted on the same device and environment as our model for a fair comparison. The top two results in each category are marked with bold and underline.
et al. [22] spired by Liu et al. [22] to present the discriminating performance of our model in Fig. 6. From the figure, it can be observed that $S_t$ rapidly decreases when anomalies appear in the frames. Once the abnormal objects disappear, $S_t$ increases immediately.

![Figure 5. Results of ablation studies on patch size.](image)

**Ablation studies on patch size.** Fig. 5 shows the result of ablation experiments that we conducted on the Avenue [24] and Ped2 [30] to observe the effect of the patch size. The patch size determines the smallest unit to be focused on by the AE. In all experiments of these ablation studies, only the size of the patch is changed between $30 \times 30$, $40 \times 40$, $60 \times 60$, and $90 \times 90$, while the frame resolution remains fixed at $240 \times 360$. It means that a comparably small region is captured in a $C_F$ with the size of $30 \times 30$, and a large region is captured in a $C_F$ with the size of $90 \times 90$. Our network shows the lowest accuracy when the patch size is $90 \times 90$, which is more than 10 percent of the frame size. When the patch is considerably large, the model focuses on larger movements than smaller ones. Abnormal conditions usually occur in small parts, hence, lower performance is observed in this case.

**Qualitative results.** We demonstrate the frame-level detecting performance of our model in Fig. 6. From the figure, it can be observed that $S_t$ rapidly decreases when anomalies appear in the frames. Once the abnormal objects disappear, $S_t$ increases immediately.

Furthermore, the pixel-level detecting capacity is observed in Fig. 7. We present examples of predicted frames and the corresponding difference maps. Additionally, we emphasize the results by comparing each sample with those

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>83.2</td>
<td>72.1</td>
<td>95.7</td>
</tr>
<tr>
<td>TMT</td>
<td>83.0</td>
<td>72.2</td>
<td>95.1</td>
</tr>
<tr>
<td>SRT</td>
<td>85.0</td>
<td>72.4</td>
<td>96.0</td>
</tr>
<tr>
<td>TMT ∪ SRT†</td>
<td>84.5</td>
<td>72.1</td>
<td>95.2</td>
</tr>
<tr>
<td>TMT ∪ SRT</td>
<td>85.3</td>
<td>72.2</td>
<td>96.3</td>
</tr>
<tr>
<td>TMT ∩ SRT</td>
<td>84.6</td>
<td>72.2</td>
<td>96.2</td>
</tr>
</tbody>
</table>

Table 2. We demonstrate the impact of our patch anomaly generation by ablation studies on CUHK Avenue [24], ShanghaiTech (ST) [28], and Ped2 [30]. We present frame-level AUC (%) of experiments on 5 variations: using only TMT, using only SRT, randomly selecting between TMT and SRT (indicated as SRT†), randomly selecting between TMT and SRT, and using both the TMT and SRT.
Figure 6. Score plot from evaluation. The red and blue lines denote $S_t$ and labels, respectively. Labels are 0 when frames are abnormal. (A) is obtained from Avenue [24]. Running, throwing a bag, and moving in the wrong direction are well detected. (B) is obtained from ShanghaiTech [28]. Chasing and running are detected as anomalies. (C) is obtained from Ped2 [30] where the captured anomalies are bicycles and a car.

of our baseline model. In the example of Ped2 [30], the bicycle is the annotated anomaly, which is an unseen appearance. In Avenue [24] and ShanghaiTech [28], the annotated anomalies relate to motion: a man throwing a bag and a running person. The outputs generated by our model trained with the patch anomaly generation phase are significantly much blurrier than those of the baseline, validating the effectiveness of our transformation phase. Note that our model nearly erased the bag and the person in the examples of Avenue [24] and ShanghaiTech [28]. This proves that our model does not simply infer abnormal objects by copying from the inputs, which is what the baseline model does. Moreover, for the ShanghaiTech dataset [28], the difference map of our model shows a distinction in a larger region compared to that of the baseline. We observe that our model did not accept the motion in the input; it attempted to predict the trajectory of the runner as it as per the training. However, the baseline model generated a moderate copy of the input based on the given trajectory.

5. Conclusion and Future Work

In this paper, we proposed a prediction network for video anomaly detection combined with a patch anomaly generation phase. We designed a light-weight AE model to learn the common spatio-temporal features of normal frames. The proposed method generated transformed frame cuboids as inputs, by applying SRT or TMT to a random patch cuboid within the frame cuboid. Our model was encouraged to pay attention to the appearance and motion patterns of normal scenes. In addition, we discussed the impact of the patch anomaly generation by conducting ablation studies. Furthermore, the proposed method achieved competitive performance on three benchmark datasets and performed at a very high speed, which is as important as the detection capacity in anomaly detection.

Through the experimental results, we also have shown that our network is able to localize the anomalies. Since detecting temporal-wise anomalies is the most essential part and there is an inconsistency issue in pixel-level AUC evaluation [37], we only considered the frame-level detection. With the newly proposed TBDR and RBDR metrics [36], our future work will be verifying the fast localizing capability of our network.
References


[55] Jia-Xing Zhong, Nanman Li, Weijie Kong, Shan Liu, Thomas H Li, and Ge Li. Graph convolutional label noise