ADVANCED INTELLIGENT SYSTEMS
Open Access
www.advintellsyst.com

Video Anomaly Detection Utilizing Efficient Spatiotemporal Feature Fusion with 3D Convolutions and Long Short-Term Memory Modules

*Sareer Ul Amin, Bumsoo Kim, Yonghoon Jung, Sanghyun Seo, and Sangoh Park**

Surveillance cameras produce vast amounts of video data, posing a challenge for analysts due to the infrequent occurrence of unusual events. To address this, intelligent surveillance systems leverage AI and computer vision to automatically detect anomalies. This study proposes an innovative method combining 3D convolutions and long short-term memory (LSTM) modules to capture spatiotemporal features in video data. Notably, a structured coarse-level feature fusion mechanism enhances generalization and mitigates the issue of vanishing gradients. Unlike traditional convolutional neural networks, the approach employs depth-wise feature stacking, reducing computational complexity and enhancing the architecture. Additionally, it integrates microautoencoder blocks for downsampling, eliminates the computational load of ConvLSTM2D layers, and employs frequent feature concatenation blocks during upsampling to preserve temporal information. Integrating a Conv-LSTM module at the down- and upsampling stages enhances the model's ability to capture short- and long-term temporal features, resulting in a 42-layer network while maintaining robust performance. Experimental results demonstrate significant reductions in false alarms and improved accuracy compared to contemporary methods, with enhancements of 2.7%, 0.6%, and 3.4% on the UCSDPed1, UCSDPed2, and Avenue datasets, respectively.

1. Introduction

Video anomaly detection pertains to the task of identifying uncommon occurrences within videos. This field has garnered considerable attention due to the widespread deployment of video surveillance systems aimed at enhancing public safety. Despite their prevalence, these surveillance systems often fall short in terms of monitoring efficacy. This discrepancy arises from the infrequent nature of abnormal events in comparison with routine incidents. Consequently, there is a growing demand for automated anomaly detection systems to alleviate the burden of continuous monitoring. However, the complexity of this task is noteworthy, primarily due to the scarcity of appropriate datasets. Compounding the issue is the variability in defining abnormal events, contingent upon the contextual specifics of each video.

A notable hurdle in anomaly detection pertains to data imbalance. This phenomenon denotes the inherent difficulty in capturing instances of abnormal scenes in contrast to their normal counterparts, primarily due to their rarity in real-world scenarios. Consequently, securing datasets that feature an equitable distribution of both categories of scenes proves challenging. Consequently, training data predominantly comprise normal videos.[1] This predicament complicates the training of models via conventional supervised approaches reliant on manually annotated data. The challenge stems from the unbounded nature of anomalies, rendering it impractical to catalogue and gather all potential abnormal events. Furthermore, the process of labeling such events is immensely labor-intensive. Thus, the successful detection of unprecedented and undefined anomalous events necessitates a system capable of learning normalcy through exposure to copious, easily accessible normal videos.

With the rise of deep learning, there has been a substantial surge in research dedicated to surveillance anomaly detection, particularly when employing normally trained videos. Unsupervised learning methodologies, predominantly centered around frame reconstruction or prediction-based techniques, have gained prominence.[2,3] A prevailing choice within these approaches involves employing autoencoder (AE) architectures, which are designed to learn reconstruction or prediction tasks

S. Ul Amin, S. Park
Department of Computer Science and Engineering
Chung-Ang University
Seoul 06974, South Korea
E-mail: sopark@cau.ac.kr

B. Kim, S. Seo
College of Art and Technology
Chung-Ang University
Anseong 17546, South Korea

Y. Jung
Department of Advanced Imaging Science Multimedia & Film
Chung-Ang University
Seoul 06974, South Korea

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aisy.202300706.

DOI: 10.1002/aisy.202300706

exclusively from normal scenes. During testing, these networks struggle to accurately reconstruct abnormal scenes, leading to substantial disparities between input and output and thus facilitating anomaly detection through significant reconstruction errors. This strategy permits training without reliance on labeled data and has exhibited considerable performance enhancements.

Within a surveillance framework, distinguishing anomalous events from normal ones is often contingent on visual characteristics, motion patterns, or a combination thereof. For instance, the presence of atypical objects like cars traversing a pedestrian sidewalk is characterized by distinct appearances compared to standard scenes. Likewise, altercations or pursuits involving individuals entail variations in motion, while instances of individuals hurling unconventional objects showcase disparities in appearance and motion. The extraction of features encompassing appearance and motion from input videos constitutes a critical aspect of anomaly detection. Numerous AE-based methodologies incorporate precise motion data, often utilizing pretrained algorithms such as optical flow,[2,4] or pose estimators.[5] Nonetheless, the explicit separation of information imparts inherent biases onto the network, tethering it to the characteristics of the pretrained model. This can potentially limit network capacity due to the influence of strong priors and hinder the complete exploitation of comprehensive spatiotemporal patterns.

Hence, we suggest an innovative methodology that attends to appearance and motion information. Our approach employs a technique termed "double encoding" through the utilization of micro-AEs, coupled with a deliberate delay during the decoding phase, achieved using residual connections akin to those found in ResNet.[6] This delay facilitates the recuperation of lost features. To address both localized and global long–short-term spatiotemporal motion dynamics inherent in surveillance videos, our method incorporates 3D convolutional layers and long short-term memory (LSTM) units. The focal contributions of our research can be outlined as follows: 1) The researchers present an innovative approach termed "double-encoding" through the incorporation of autoencoder-like micro modules and a gradual decoding process facilitated by feature-passing residual connections. In this context, a given input feature map at a particular stage of the downsampling procedure undergoes dual encoding before eventually reaching the subsequent level of the lower dimensional map. During upsampling, decoding is performed using residual feature flows from downsampling stages for each new feature space dimension. 2) To address the temporal dynamics in time-dependent video sequences, the methodology incorporates 3D convolutions, which are adept at capturing brief temporal motions. Concurrently, for the intricate interplay between long- and short-term temporal motions, LSTM modules are strategically integrated. These LSTM modules operate within the down- and upsampling stages, serving to capture these temporal dynamics effectively. 3) The proposed technique is rigorously evaluated through experiments conducted on challenging datasets known for their complexity—UCSDPed1,[7] UCSDPed2,[7] and CUHK-Avenue.[8] The experimental results substantiate the efficacy of our strategy, showcasing its effectiveness over existing techniques based on standard evaluation metrics.

The following sections are structured in the following manner. A brief overview of related work is presented in Section 2. Section 3 provides an in-depth discussion of the proposed techniques. Results of the experiment with a comprehensive analysis are presented in Section 4. Finally, the conclusion and future directions of this article are outlined in Section 5.

2. Related Work

The literature pertaining to methods for anomaly detection can be classified into two primary groups: those based on convolutional networks and those focused on frame reconstruction, both of which are utilized for identifying anomalous events. In the domain of video data, 3D convolution-based networks serve as fundamental architectures for extracting features in video anomaly detection. Recent explorations into video representation learning encompass transformer-based models.[9–12] Notably, the TimeSformer[9] and Video Vision Transformer (ViViT)[10] models segment spatial and temporal information to accomplish self-attention through a series of spatiotemporal patterns. The Multiscale Vision Transformers (MViT)[11] integrate multiple stages of channel-resolution scales, culminating in a multiscale pyramid of feature maps. Additionally, the Swin-T[12] also extends the Swin-T paradigm to the spatiotemporal domain, enhancing the modeling of videos' spatiotemporal locality. These approaches frequently employ Resnet backbones due to their efficiency. In video recognition task, it has been suggested to represent motion aspects using two-stream networks explicitly. These networks require the specific extraction of temporal details, such as optical flows or temporal discrepancies. Within the unsupervised learning paradigm for representation learning, Siamese network structures with contrastive learning[13] have garnered attention. These structures try to reduce the similarity of negative pairings while increasing that of positive pairs. Additionally, the MvCLN method[14] facilitates consistent representation learning from multiview/modal data. Recent advancements in video recognition have yielded the "slow and fast pathway" networks, as proposed by Feichtenhofer et al.[15] This architecture effectively extracts both static and motion details from distinct temporal and spatial dimensions. The "fast pathway" operates at a higher temporal rate than the "slow pathway" which employs a narrower temporal window. In contrast, several anomaly detection algorithms centered on frame reconstruction leverage the robust representational capabilities inherent in deep convolutional networks. These algorithms capitalize on the architecture of convolutional AEs,[16] recurrent neural networks (RNNs),[17] or 3D convolutions.[18] Alternatively, some algorithms adopt different reconstruction objectives for learning, harness memory modules,[19,20] reconstruct optical flows from video sequences,[21] or encode regular patterns through sparse dictionary learning.[22,23] A notable example is the sparse LSTM unit introduced by Zhou et al., which uses adaptive ISTA L1-solvers to store historical data for unsupervised anomaly detection.[22]Similarly, the temporally coherent sparse coding framework by Luo et al. introduces a specialized type of sRNN to maintain coherence across adjacent frames in videos.[23] Predictive methods relying on frame prediction have also emerged to heighten the unpredictability of abnormal samples.[2,24,25] However, these prediction-based approaches typically necessitate more substantial architectures or optical flows, and

bidirectional methods often depend laboriously on future frames. Another approach to anomaly detection uses methods to learn about the normal clusters' compactness. These clusters can be created by either excluding little clusters of typical samples and their related characteristics from the reconstruction objective[26] or by extracting clusters from pretrained detectors[27] or pose estimators.[5] In contrast to the existing methodologies, this article introduces a streamlined approach that intrinsically prioritizes appearance and temporal details for video anomaly detection. The presented method effectively harnesses the features of 3D Convo-STM to manage both local and global long–short-term spatiotemporal motion information within surveillance videos. The key characteristics of this innovative approach are elaborated upon in Section 3.

3. Methodology

The proposed model employs a strategic combination of 3D convolutions and LSTM modules, facilitating structured coarse-level feature fusion. This framework is visually demonstrated in **Figure 1**. The incorporation of micro-AE blocks is rooted in the principle of enhancing network depth rather than broadening it, ultimately leading to improved feature generalization. To address the vanishing gradient problem often encountered in deep networks, residual feature flows are introduced. These flows serve to transport crucial information from earlier layers to later ones. This strategy not only simplifies training but also reduces the total parameter count.[6] A diagram in **Figure 2**a represents a ResNet connection, where X is an input feature, $H(X)$

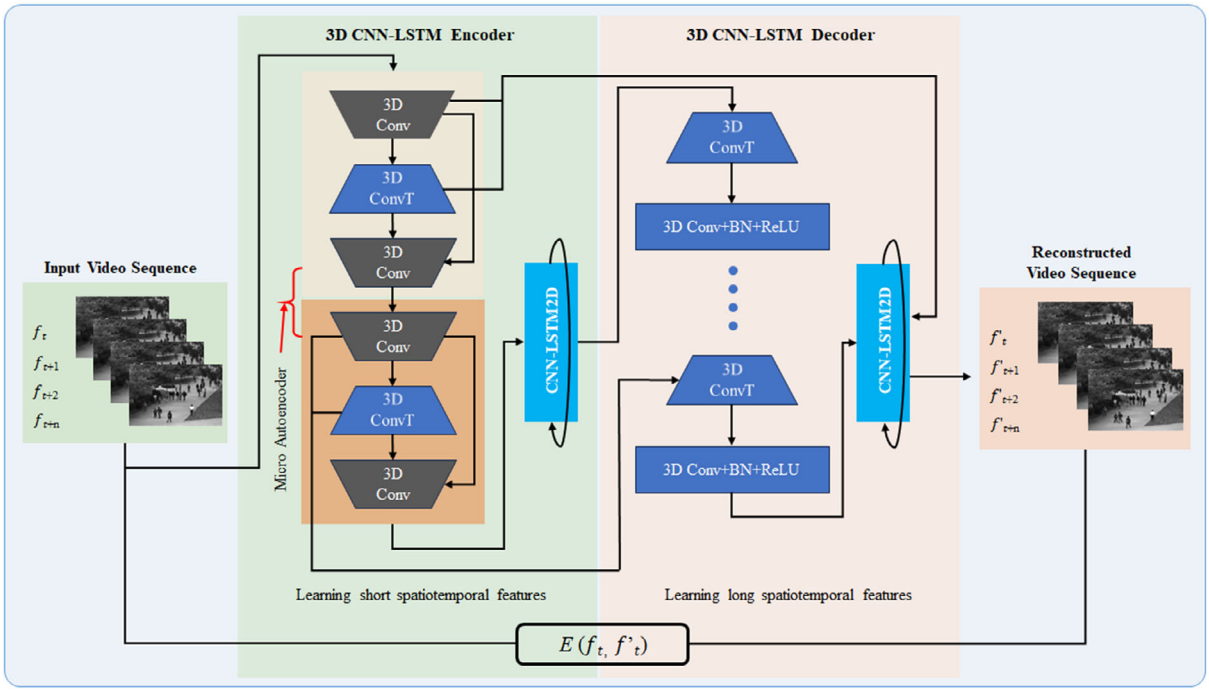

Figure 1. The proposed method integrates micro-AE blocks during downsampling, eliminates the computational load of ConvLSTM2D layers, and introduces frequent feature concatenation blocks during upsampling to retain temporal information. The incorporation of a Conv-LSTM module at the end of the down- and upsampling stages enhances the model's capacity to capture long–short-term temporal features.

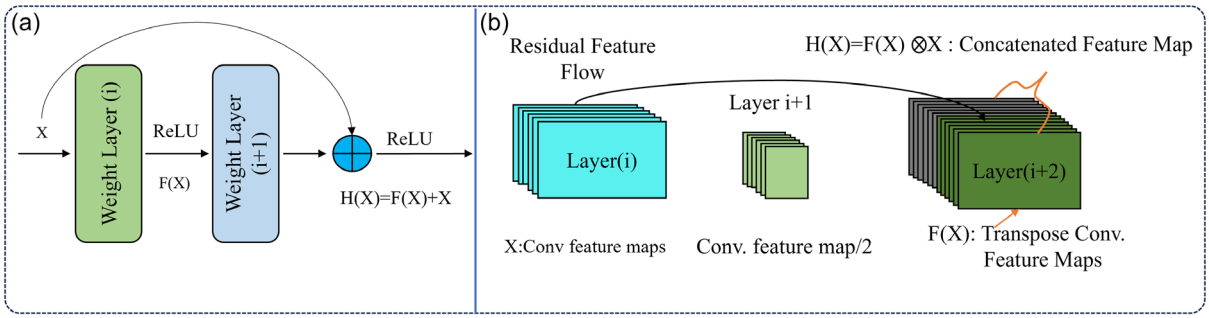

Figure 2. a) A ResNet connection, where X is an input feature, $H(X)$ is the expected transformation, and $F(X)$ is the residual modeling. b) The features are concatenated at a coarse level and stacked depth-wise.

is the expected transformation, and $F(X)$ is a residual modeling. In the context of ref. [6], the feature fusion operation $H(X) = F(X) + X$ is achieved using a shortcut connection and element-wise addition. However, in our model, the features are concatenated at a coarse level and stacked depth-wise, as shown in Figure 2b. This approach leads to a reduction in the number of filters within convolutional layers, subsequently reducing computational demands.

Furthermore, **Figure 3** provides a comprehensive depiction of the 3DCNN-LSTM model. It outlines the specific operations undertaken in each layer, the corresponding activation functions, and the resultant output dimensions. The architecture has been optimized to strike a balance between parameter count and performance. The design reduces the computational load of ConvLSTM2D layers by introducing three micro-AE blocks within the downsampling process. On the other hand, multiple feature concatenation blocks are used to actively pace the upsampling process. A convolutional LSTM (Conv-LSTM) module is incorporated after the down- and upsampling phases to provide long–short-term temporal patterns. Consequently, the proposed method attains a depth of 42 layers, rendering it more robust and effective.

3.1. 3D Conv-LSTM

The 3D convolution operation holds significance in the realm of spatiotemporal representation learning.[28] Unlike the 2D convolutional layer, which operates exclusively in the spatial domain, 3D convolution operates across both spatial and temporal dimensions. As a result, 3D convolution carries out convolutional operations that cover both space and time, allowing it to extract local or short temporal key features.

Subsequently, these extracted local temporal features are funneled into Conv-LSTM units. These units are designed to sustain long-term temporal connectivity between consecutive frames. In essence, the 3D convolution operation enables the extraction of short-term temporal characteristics, while the Conv-LSTM units facilitate the retention of temporal dependencies over extended time frames. The effectiveness of the Conv operation hinges on its filter weights, which are iteratively updated during training. A basic 2D Conv C output feature map *w.r.t.* kernel omega and an input image/patch x are calculated as

$$C(m,n) = \sum_{k=0}^{K-1}\sum_{l=0}^{K-1} \omega(k,l) * x(m+k, n+l) \tag{1}$$

In the given context, the following notations are employed: $*$ represents the convolution operation; K signifies the size of the kernel; $\{m,n\}$ denotes the image or patch's origin or initial coordinate; and $\{k,l\}$ refers to the specific position of an element within the kernel.

A convolutional layer's activation map dimension is determined using the following formula

$$\left(\frac{I_s - K_s + 2 \times P}{S}\right) + 1 \tag{2}$$

where I_s represents the size of a patch or image; K_s signifies the filter size; P signifies the number of zero-padded pixels; and S is the stride rate.

Expanding upon this, the 3D convolution ($C3D$) operation is calculated in the manner shown below

$$C3D(q,m,n) = \sum_{t=0}^{T-1}\sum_{k=0}^{K-1}\sum_{l=0}^{K-1} \omega(t,k,l) * x(q+t, m+k, n+l) \tag{3}$$

where $*$ represents the convolution operation; T denotes the temporal length of the data; K signifies the size of the kernel; $\{q,m,n\}$ represents the first coordinate or origin of the input patch; and $\{t,k,l\}$ denotes the element index of the kernel.

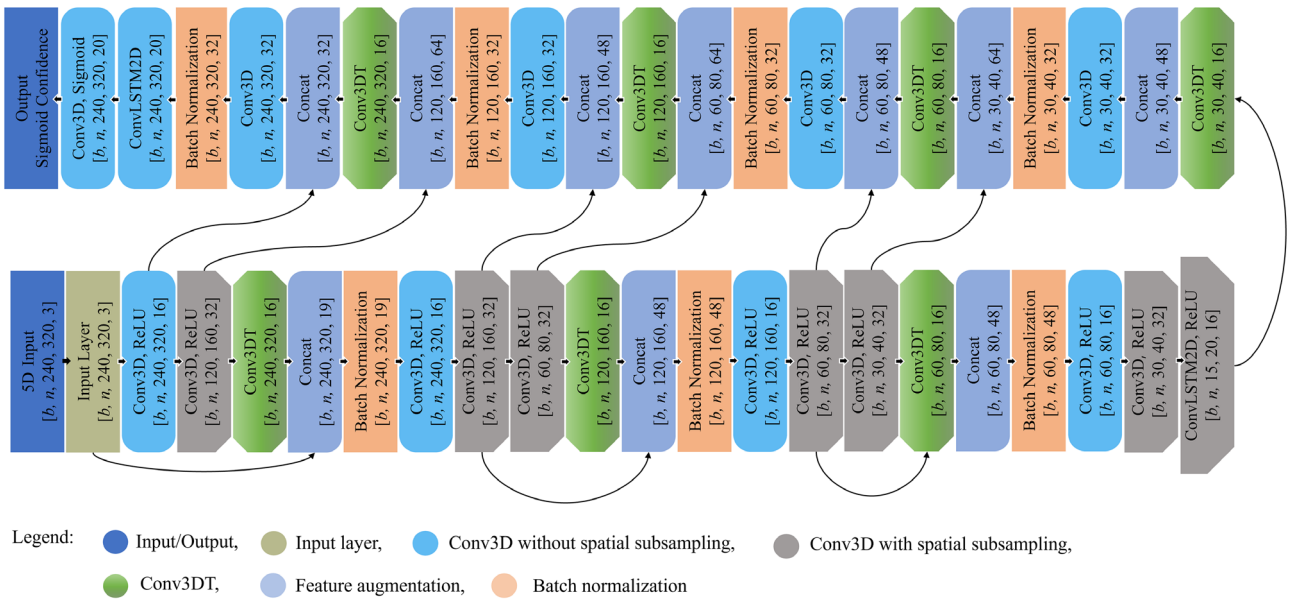

Figure 3. A 3D CNN-LSTM framework is illustrated layer-by-layer employing AE-like micromodules and a slow-decoding approach.

Consequently, conventional 1D LSTMs account for temporal dependencies but do not encompass spatial dependencies. In contrast, this work integrates 2D LSTMs with 3D convolution, thereby incorporating both spatial and temporal relationships. The standard LSTM unit can be seen in **Figure 4**. The inputs are denoted by $X_1, \ldots, X_t$, while C_t represents the cell state, H_t signifies the hidden state, and it, ft, and ot are the input, forget, and output gates of a ConvLSTM block. Using $*$ and $\circ$ to represent the convolution operator and the Hadamard product, respectively, the ConvLSTM block's computation can be explained as follows:

$$it = \sigma(W_{xi} * X_t + W_{hi} * H_{t-1} + b_i) \tag{4}$$

$$ft = \sigma(W_{xf} * X_t + W_{hf} * H_{t-1} + b_f) \tag{5}$$

$$ot = \sigma(W_{xo} * X_t + W_{ho} * H_{t-1} + b_o) \tag{6}$$

$$Ct = ft \circ Ct - 1 + it \circ \tanh(W_{xc} * X_t + W_{hc} * H_{t-1} + b_c) \tag{7}$$

$$Ht = ot \circ \tanh(Ct) \tag{8}$$

where W_{xi} and W_{hi} are the spatial dimensions of convolution filters, and σ is a hard sigmoid function working as the recurrent activator.

3.2. Transpose Convolution

The transpose convolution layers execute an upsampling operation on the 3D convolution, effectively doubling the spatial sizes of the output feature maps compared to the input while retaining the connectivity pattern. Unlike spatial resizing, which requires extrapolation, the transpose layer incorporates trainable parameters. This process involves inserting zeros between adjacent neurons in the input receptive field, followed by sliding the convolution filter with unit strides.[29]

3.3. Activation Functions

Activation functions play a crucial role in enhancing the representation capabilities of neural networks (NNs) by introducing nonlinear components. This is especially vital as the linear representation of convolution operations reaches its limitations within deep architectures.

The commonly used rectified linear unit (ReLU) activation function is described as follows

$$y_k = \max\left(0, \mathbf{w}_k^T \mathbf{x}\right), \quad \text{for} \quad k = 1, 2, \ldots, K \tag{9}$$

Here, $\mathbf{w}_k \in \mathrm{R}^N$ represents anchor vectors with K possible instances, and $\mathbf{x}$ is the input. This formulation introduces nonlinear rectification to the output $\mathbf{y} = (y_1, \ldots, y_K)^T$. ReLU effectively clips negative values to zero while leaving positive values unaltered. ReLU offers advantages such as sparsity, mitigation of vanishing gradient issues, and computational efficiency compared to other activation functions.

On the other hand, sigmoid is another activation function with an output range of [0, 1] for any given input x, defined by

$$f(x) = \frac{1}{1 + \exp(-x)} \tag{10}$$

Sigmoid is suitable for binary classification tasks and linear regression concerns. Meanwhile, a linear piece-wise function, called the hard sigmoid, approximates outputs by linearly interpolating between two cut points. This activation function is extremely efficient in terms of computing.[30]

3.4. Batch Normalization

The mathematical definition of batch normalization (BN) is described below. Consider the layer's output, denoted by $X \in \mathrm{R}^{N,D}$, where N is the sample count in the mini-batch and D is the number of neurons in the hidden layer. Then, the normalized matrix $\hat{X}$ is computed as

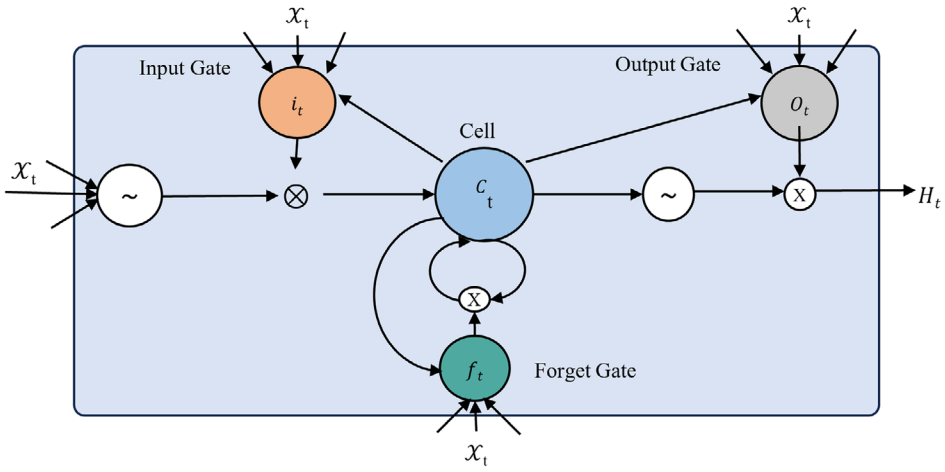

Figure 4. A typical LSTM module is made of three gates: an input gate, a forget gate, and an output gate. These gates coordinate to manage the information flow throughout the module.

$$\hat{X} = \frac{X - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}} \tag{11}$$

Here, μ_B, σ_B^2, and ε correspond to the mini-batch's mean, variance, and a small value (0.001), respectively, introduced to discourage division by zero. Subsequently, the strength of the layer's representation is maintained by employing the identity transformation

$$Y = \gamma\hat{X} + \beta \tag{12}$$

Here, β and γ are trainable parameters initialized as $\beta = 0$ and $\gamma = 1$ in this context. It is noteworthy that when β is set to μ_B and γ is set to $\sqrt{\sigma_B^2 + \varepsilon}$, the outcome mirrors the activation map of the previous layer.

The adoption of BN yields several advantages: 1) It mitigates internal Covariate shift by maintaining μ_B and σ_B closer to zero and one. 2) During training, normalizing the batch of samples can improve the model's generalization ability. 3) Placing BN before nonlinearities prevents issues like training saturation in nonlinear regions, effectively addressing concerns about vanishing gradients. 4) It facilitates faster training with higher learning rates while minimizing the need for meticulous initialization.[31]

3.5. Training Loss

The reconstruction loss of the presented framework guides the model to learn meaningful representations of the input data and produce reconstructions that closely resemble the original input. The reconstruction loss of the proposed method can be represented mathematically as the mean squared error between the input tensor x and the reconstructed output tensor

$$L_{rec} = \frac{1}{HW}\sum_{c=1}^{3}\sum_{h=1}^{H}\sum_{w=1}^{W}\left(x_{c,h,w} - y_{c,h,w}\right)^2 \tag{13}$$

where

c represents the channel dimension (1–3 for RGB channels), h represents the height dimension (1 to H), w represents the width dimension (1 to W), and $x_{c,h,w}$ represents the value of the element at channel c, height h, and width w in the input tensor x, while $y_{c,h,w}$ represents the value of the element at channel c, height h, and width w in the reconstructed output tensor. However, H and W represents the total number of pixels in each channel ($H \times W$).

The reconstruction loss represents the dissimilarity between the input tensor and the reconstructed tensor, and the goal is to minimize this loss during training. This loss term measures how well the proposed method can reconstruct the original input from its learned representations. During training, the aim is to minimize loss, encouraging the model to capture and encode meaningful features from the input data, ultimately improving reconstruction quality.

4. Experimental Results and Discussion

This research assesses the suggested framework's effectiveness using three benchmark datasets: CUHK-Avenue,[8] UCSDPed1,[32] and UCSDPed2.[33] **Table 1** presents comprehensive details on video count, dataset splitting ratio, and kinds of abnormalities, while **Figure 5** showcases sample frames of both usual and unusual events. The proposed approach is quantitatively evaluated employing two metrics: area under the curve (AUC) and receiver operating characteristic (ROC).[7]

4.1. Experimental Setting

The suggested method was executed in Keras with TensorFlow as its backend, using a Python version 3.9 programming environment. The results were conducted on a personal computer rigged with an NVidia GPU (3070 RTX) having 32 GB of RAM, running on the Windows 10 OS and the CUDA toolkit version 11.0 along with cuDNN v8.0. The quantitative results showed that our proposed framework was highly effective and outperformed the contemporary approaches by a substantial margin.

4.2. Implementation Details

This study used the well-known deep learning framework, specifically the Keras-backed TensorFlow module, while using Python programming language. To ensure a fair comparison, we used existing methods previously trained on considerable UCSDPed1, UCSDPed2, and Avenue datasets. To improve the convergence of our presented method, we shifted significant hyperparameters. Our rigorous experimentation has identified the optimal learning rate (0.001) and dropout value (0.5) to achieve the best performance. We also trained the proposed method using Adam, SGD, and Adadelta optimization algorithms, but the Adadelta optimization algorithm performed the best. The suggested method is optimized to handle a batch size of 16 with input dimensions of 240×320. During training, we kept the number of epochs for each dataset to 50. We used a dropout of 0.5 between the last two layers for the proposed method on UCSDPed1, UCSDPed2, and CUHK-Avenue as displayed in Figure 3. The graphs in **Figure 6** show the performance attained by these hyperparameters to make a fair comparison with current techniques.

Table 1. Statistical details of the benchmark video anomaly detection datasets.

Dataset	No. of videos	Training set	Testing set	Dataset length [min]	Example of anomalies
UCSDPed1[32]	70	34	36	5	Bikers, carts, mini truck, etc.
UCSDPed2[33]	28	16	12	5	Bikers
CUHK-Avenue[8]	38	16	21	5	Run, toss, and discover a new object

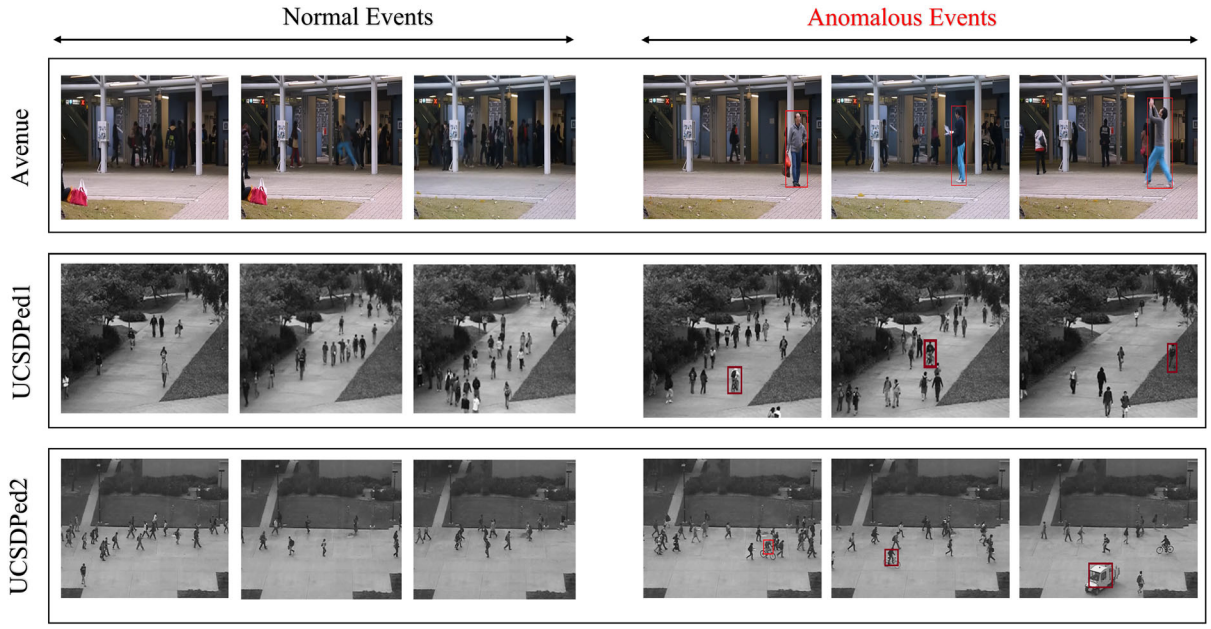

Figure 5. Sample images of the anomaly detection datasets.

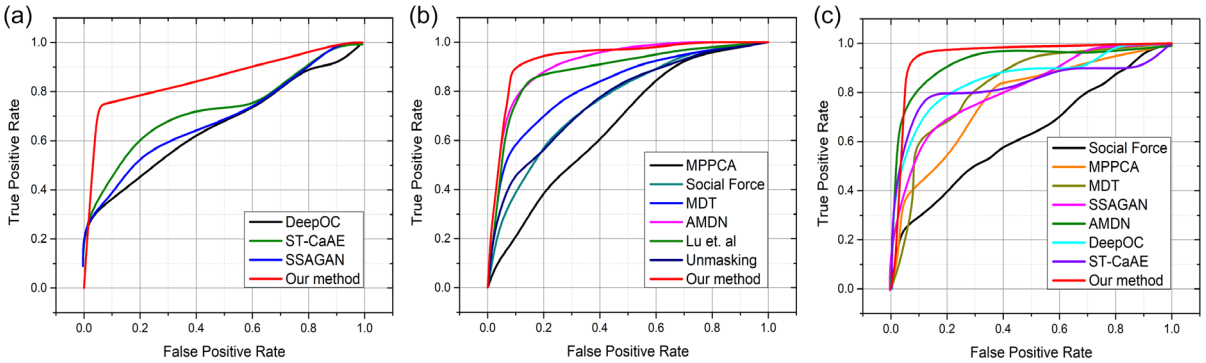

Figure 6. A comparison was made between the performance of the suggested technique and the contemporary techniques, utilizing a ROC evaluation matrix. The ROC curve of the suggested method demonstrates a higher margin, highlighted in red. The ROC curves for the CUHK-Avenue, UCSDPed1, and UCSDPed2 datasets are presented as (a–c), respectively.

4.3. Datasets

A stationary video camera with a 640 × 360 pixel resolution was used to shoot the CUHK-Avenue dataset.[8] The dataset consists of 21 test video clips that reveal aberrant human behavior and 16 training clips that show typical human activity. Walking on the sidewalk and gatherings of people on the pavement are two instances of routine behavior. Contrarily, abnormal behavior includes people dropping things, loitering, coming close to the camera, strolling on the grassland, and dumping objects.

The UCSDPed1 dataset, documented in an article cited as ref. [32], consists of 14 000 frames spread across 70 video sequences. It is divided into a training set comprising 34 video sequences and a test set comprising 36. The dataset includes 40 abnormal events, such as small carts, bikers, and mini trucks, making it a useful benchmark for anomalous occurrence detection in surveillance video.

The UCSDPed2 dataset, sourced as,[33] comprises 4560 frames from 28 different videos. It is grouped into a training set with 16 video sequences and a test set with 12 video sequences, particularly concentrating on unusual events, including bikers. As there are 12 instances of these kinds of occurrences in the dataset, it is a useful resource to assess algorithms designed to identify unusual bicycle-related events.

4.4. Applications

Anomaly detection is an essential feature of monitoring systems employed across various domains, assisting as an early-stage defense regarding security breaches, fraudulent activity, and possible risks. Anomaly detection algorithms play a significant role in security surveillance[34] by detecting suspicious activity or intrusions and allowing for prompt responses to stop security breaches and protect vital assets. Similarly, anomaly detection in

financial surveillance systems[35] is essential for identifying fraudulent activity or transactions by examining patterns in financial transactions, securing the assets of clients, and preserving the credibility of financial organizations. Furthermore, in traffic surveillance,[36] anomaly detection is crucial in tracking and controlling traffic flow smoothly by detecting unusual traffic trends or accidents in real time. This allows officials to take preventative steps to reduce traffic jams while enhancing road safety. Anomaly detection in healthcare surveillance[37] enables faster medical action and improves patient care by facilitating the early diagnosis of abnormal health issues through the analysis of physiological data. Moreover, anomaly detection has been essential for environmental surveillance[38] because it allows for the timely reduction of risks and public health safety by tracking environmental variables and detecting abnormalities that could indicate signs of pollution or potential environmental issues. Industrial surveillance systems improve productivity by monitoring device functionality, output standards, and energy usage, allowing for early detection of equipment breakdowns or potential risks, reducing downtime.[39]

4.5. Evaluation Metrics

The ROC curve was employed as a standard assessment parameter in prior research.[8] The ROC metric is measured by steadily adjusting the threshold of typical scores. Following that, the model's performance is quantitatively evaluated by calculating the AUC. The capacity of the model to differentiate between instances that are positive and negative is shown by the area under the ROC curve (AUC–ROC). These parameters are calculated employing the true positive rate (TPR) and false positive rate (FPR).

The ratio of accurately detected true positive events to the total number of real positive events (anomalies) is known as the TPR, which is also called sensitivity or recall.

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{14}$$

Here, TP stands for true positives, which are the number of anomalous events that the system correctly identified; FN stands for false negatives, which are the number of unusual events that the system missed.

The FPR is calculated as the ratio of incorrectly positive predictions to the total number of true negative events (nonanomalies)

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{15}$$

Here, FP stands for false positives, or the number of normal events that were mistakenly identified as anomalies, and TN for true negatives, indicating the number of normal events that were correctly identified as nonanomalies.

The ROC curve is created by graphing TPR versus FPR at various decision thresholds. Every detail on the curve represents how well the model performed at a specific threshold value. The curve usually starts at (0,0) and ends at (1,1).

A scalar metric called AUC–ROC is used to evaluate an anomaly detection model's overall efficiency. The integral of the TPR across a range of threshold values is represented by Equation (16)

$$\mathrm{AUC} = \int_0^1 \mathrm{TPR}(\mathrm{fpr})\ \mathrm{dfpr} \tag{16}$$

The integral sign $\int$ indicates the definite integral, which computes AUC–ROC, where TPR (fpr) signifies a function of the false positive rate. The integration is conducted throughout the whole FPR range of 0–1.

4.6. Avenue Dataset: Comparison with Recent Techniques

The freely available Avenue dataset is widely used to evaluate various video anomalous event detection approaches. In this study, we compared the suggested system to several methods that are currently being employed on this dataset,[2,3,8,16,17,19,20,22,40–48] including supervised and unsupervised methods, and exemplified their frame-based AUC. Contemporary methods, such as Che et al.[41] and Zhou et al.[22] showed AUC values of 89.6% and 86.1%, respectively. Furthermore, Zhou et al.[22] outperformed previous strategies, including those described in ref. [48], and reported the second-highest accuracy. However, our proposed system has demonstrated prominent performance compared to contemporary methodologies,[2,3,8,16,17,19,20,22,40–48] with a frame-based AUC value of 93% for the Avenue dataset. **Table 2** presents a quantifiable assessment of our suggested approach and other existing techniques using frame-based AUC values. The ROC curve for our suggested approach is

Table 2. Frame-Based AUC contrast of our (suggested) framework and recent approaches on UCSDPed1,[32] UCSDPed2,[33] and CUHK-Avenue[8] datasets.

Methods	UCSDPed1[32]	UCSDPed2[33]	CHUK-Avenue[8]
Lu et al.[8]	91.8	–	80.9
Radu et al.[40]	68.4	82.2	80.6
Zhou et al.[22]	83.5	94.9	86.1
Che et al.[41]	–	–	89.6
Mahmudul et al.[16]	81.0	90.0	70.2
Weixin et al.[42]	75.5	88.1	77.0
Weixin et al.[17]	–	92.2	81.7
Tang et al.[3]	82.6	96.2	83.7
Dong et al.[19]	–	94.1	83.3
Qiang et al.[43]	85.2	–	85.8
Hyunjong et al.[20]	–	90.2	82.8
Wen et al.[2]	83.1	95.4	85.1
Yao et al.[44]	84.5	95.9	85.9
Ramachandra et al.[45]	77.3	88.3	72.0
Zhang et al.[46]	94.2	92.9	80.5
Yiwei et al.[47]	86.2	96.0	85.7
Zhou et al.[48]	83.9	96.0	86.0
Proposed method	94.5	96.8	93.0

displayed in Figure 6a, outperforming current methods by a considerable margin.

4.7. UCSD Pedestrian Dataset: Comparison with Recent Techniques

The UCSD Pedestrian dataset is freely available and is widely utilized to evaluate algorithms aimed at detecting anomalies in surveillance videos. In this study, we propose a novel strategy for detecting abnormalities in surveillance videos and compare its performance with that of other existing techniques.,[2,3,8,16,17,19,20,22,40,42–48] using the UCSDPed1 and UCSDPed2 datasets. We utilized ROC and AUC as the assessment metrics and created the AUC and ROC curves employing the test set from the UCSDPed1 and UCSDPed2 datasets to assess the effectiveness of our suggested system. Subsequently, we compared our results to those reported in current methods.[2,3,8,16,17,19,20,22,40,42–48] Table 2 compares the frame-based AUC values for UCSDPed1 and UCSDPed2 datasets with other contemporary techniques. The results reveal that the presented strategy exceeds the existing methods, as it achieved the highest frame-based AUC value of 94.5% for the UCSDPed1 dataset. Moreover, the studies in refs. [8,46,47] reported AUC values of 91.8%, 94.2%, and 86.2%, respectively, indicating that the suggested approach is highly effective on the UCSDPed1 dataset.

According to Table 2, the outcomes of our experiments on the UCSDPed2 dataset exhibit that our presented framework reveals an effective performance compared to other contemporary techniques.[2,3,8,16,17,19,20,22,42–48] Our model achieved an AUC score of 96.8%, which is 0.6% higher than the most recent method.[3] Lastly, the proposed method yielded impressive results, achieving an AUC score of 96.8%, surpassing the most recent method outlined in ref. [3] by 0.6%. As shown in Figure 6b,c, our proposed framework exhibited superior performance on both the UCSDPed1 and UCSD-Ped2 datasets, outperforming recent studies by a notable margin.

4.8. Anomaly Score

An anomaly score is a numerical representation of the degree to which a particular region, frame, or event in the video deviates from what is believed to be usual or expected. The objective of surveillance video analysis is to detect odd or aberrant behavior, occasions, or items that deviate from the regular patterns of the area being watched. The anomaly scores of the suggested technique along with video frames against three datasets are shown in **Figure 7**. A blue line is drawn over the *y*-axis and frames are placed on the *x*-axis to represent the anomalous score.

An anomaly score in surveillance video anomaly detection represents a quantitative measure that indicates the degree to which a specific region, frame, or event in a video deviates from what is considered normal or expected. In the context of surveillance video analysis, the goal is to identify unusual or anomalous behavior, events, or objects that stand out from the typical patterns of the environment being monitored. Figure 7 presents anomaly scores of the proposed method along video frames from three datasets. Note that the anomaly score is drawn using a blue line across the *y*-axis and frames on the *x*-axis. The blue line varies between irregular and regular events in a particular test video, suggesting that our technique can differentiate aberrant events from a large number of regular events. The abnormality score of test video 04 in the Avenue dataset[8] is visualized in Figure 7a; the abnormality score of test video 23 in the UCSDPed1[32] dataset is displayed in Figure 7b. Anomaly occurs at the start of the video and continues until the anomaly score drops below the cutoff, demonstrating the suggested model's ability to identify anomalies successfully. Additionally, the abnormality score of test video 02 in the UCSDPed2[33] dataset is displayed in Figure 7c.

4.9. Proposed Framework Efficiency

To fully comprehend the complexity and resource requirements of our presented method, we compare its efficiency with contemporary techniques in terms of parameter count, model size, and time complexity per sequence. The proposed model's size in megabytes (MB) and parameter count, expressed in millions, are presented in **Table 3**. The proposed method has around 0.224 million learnable parameters, demonstrating its intricacy and ability to capture complex patterns in the data. Furthermore, the model's applicability is demonstrated by its compact model size of around 2.83 MB, which contributes to

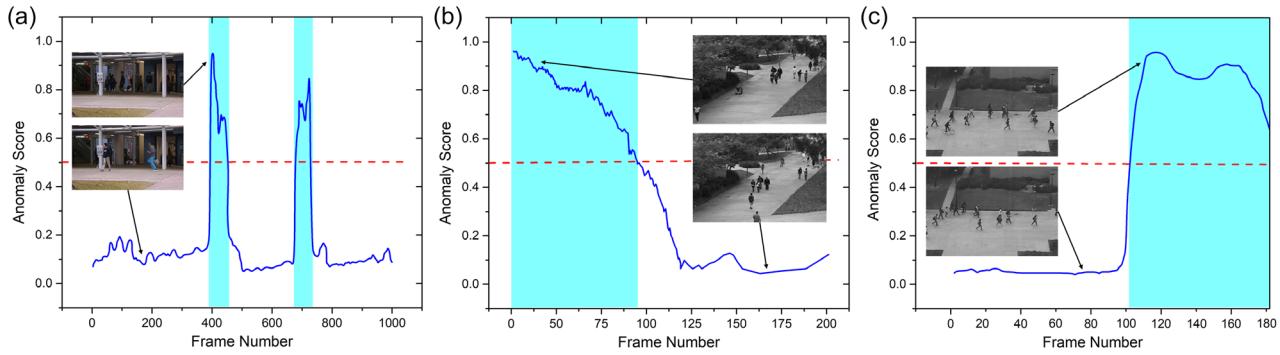

Figure 7. a) The graph shows the abnormality score on test video 04 in the Avenue dataset. b) The graph displays the abnormality score on test video 23 in the UCSDPed1 dataset. c) The graph depicts the abnormality score on test video 02 in the UCSDPed2 dataset. The graph represents the ground truth anomalous frames in cyan. To make it more understandable, we normalized the abnormality scores for each video within the range of 0–1. The graph shown above displays that as abnormalities occur, the score of the abnormalities also increases.

Table 3. Proposed framework efficiency in terms of parameter count (in millions), model size (MBs), and time complexity per sequence (ms).

Method	Parameter count [M]	Model size [MB]	Time complexity/ sequence [ms]
VGG19+ multilayer BD-LSTM[49]	143.00	605.50	220
Inception V3+ multilayer BD-LSTM[50]	23.00	148.50	180
ResNet-50 + multilayer BD-LSTM[51]	25.00	143.00	200
TransCNN[52]	26.06	122.1	–
EADN[53]	14.14	53.90	200
Proposed method	0.224	2.83	160

a considerably low time complexity of just 160 ms, as it requires only 160 ms to process a sequence of 30 frames. These performance measures highlight the balance between model complexity and resource efficiency, which makes our model a promising alternative for detecting anomalies in surveillance videos.

5. Conclusion and Future Direction

Intelligent surveillance systems are essential for security, enabling quick responses to anomalies in surveillance scenes. However, these systems demand substantial data and robust processing capabilities. In this study, we introduce a deep learning-based approach for video abnormality detection that outperforms existing methods using benchmark datasets. The proposed approach combines 3D convolutions and LSTM modules to capture spatiotemporal features effectively. A key innovation is the structured coarse-level feature fusion mechanism, which promotes feature generalization and mitigates vanishing gradients. Unlike traditional convolutional neural networks (CNNs) that use shortcut connections and element-wise addition for feature fusion, our approach stacks features depth-wise, reducing computational complexity and optimizing the model architecture. Furthermore, we illustrate the ResNet-inspired connection mechanism, highlighting the distinction of our feature fusion technique. Our model balances efficiency and performance, with a detailed architecture for the 3DCNN-LSTM model, including operations, activation functions, and output dimensions. The model incorporates micro-AE blocks for efficient downsampling and frequent feature concatenation blocks for robust upsampling. The integration of Conv-LSTM modules enhances its ability to capture long–short-term temporal features, making it a deep and robust 42-layer network. Experimental results demonstrate that our approach significantly reduces false alarms and improves accuracy, achieving 2.7%, 0.6%, and 3.4% enhancements on the UCSDped1, UCSDped2, and Avenue datasets, respectively. However, we acknowledge the necessity of improving real-time accuracy and efficiency in spotting anomalies in video surveillance. Considering the proposed method's difficulties, especially when dealing with challenging environments and unbalanced classes, we focus on the significance of enhancing its potential to work in real time while preserving optimal accuracy. This requires further advancing the methodology to mitigate false negatives while boosting the speed of processing and facilitating prompt and reliable anomaly detection in surveillance videos. The intricacy of scenes, encompassing diverse objects, motions, changes in brightness, and occlusions, presents noteworthy challenges for detecting anomalies. Our future work aims to incorporate a spatial–temporal attention block and further assess additional anomaly detection datasets to enhance our architecture's precision and overall efficiency.

Acknowledgements

This study supported by Culture, Sports and Tourism R&D Program through the Korea Creative Content Agency (KOCCA) grant funded by the Ministry of Culture, Sports and Tourism (MCST) in 2023 (Project Name: Development of digital abusing detection and management technology for a safe Metaverse service, Project Number: RS-2023-00227686, Contribution Rate: 100%) and the Korea Institute for Advancement of Technology (KIAT) grant funded by the Korea Government (MOTIE) (P0020632, HRD Program for Industrial Innovation).

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

S.Ul.A.: Conceptualization, methodology, visualization, writing—original draft. B.K.: Visualization, writing—review and editing. Y.J.: Visualization, writing—review and editing. S.S.: Conceptualization, methodology, supervision, investigation, writing—review and editing. S.P.: Conceptualization, methodology, writing—review and editing.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Keywords

anomaly detection, deep learning, feature fusion, finetuning, intelligent surveillance video analysis

Received: October 30, 2023
Revised: May 13, 2024
Published online: June 19, 2024

[1] V. Chandola, A. Banerjee, V. Kumar, *ACM Comput. Surv.* **2009**, *41*, 1.
[2] W. Liu, W. Luo, D. Lian, S. Gao, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2018**, pp. 6536–6545.
[3] Y. Tang, L. Zhao, S. Zhang, C. Gong, G. Li, J. Yang, *Pattern Recognit. Lett.* **2020**, *129*, 123.
[4] H. Vu, T. D. Nguyen, T. Le, W. Luo, D. Phung, in *Proc. of the Thirty-Third AAAI Conf. on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conf. and Ninth AAAI Symp. on Educational Advances in Artificial Intelligence*, AAAI'19/IAAI'19/EAAI'19, AAAI Press, Washington, DC **2019**.

[5] A. Markovitz, G. Sharir, I. Friedman, L. Zelnik-Manor, S. Avidan, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2020**, pp. 10539–10547.
[6] K.-K. Maninis, S. Caelles, Y. Chen, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, L. Van Gool, *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 1515.
[7] W. Li, V. Mahadevan, N. Vasconcelos, *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *36*, 18.
[8] C. Lu, J. Shi, J. Jia, in *Proc. of the IEEE International Conf. on Computer Vision*, IEEE, Piscataway, NJ **2013**, pp. 2720–2727.
[9] B. Gedas, W. Heng, T. Lorenzo, Is Space-Time Attention All You Need for Video Understanding, arXiv:2102.05095, **2021**, *2*, 4.
[10] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, C. Schmid, in *Proc. of the IEEE/CVF International Conf. on Computer Vision*, IEEE, Piscataway, NJ **2021**, pp. 6836–6846.
[11] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, C. Feichtenhofer, in *Proc. of the IEEE/CVF International Conf. on Computer Vision*, IEEE, Montreal, QC **2021**, pp. 6824–6835.
[12] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, H. Hu, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2022**, pp. 3202–3211.
[13] R. Hadsell, S. Chopra, Y. LeCun, in *2006 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR'06)*, Vol. *2*, IEEE, Piscataway, NJ **2006**, pp. 1735–1742.
[14] M. Yang, Y. Li, Z. Huang, Z. Liu, P. Hu, X. Peng, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2021**, pp. 1134–1143.
[15] C. Feichtenhofer, H. Fan, J. Malik, K. He, in *Proc. of the IEEE/CVF International Conf. on Computer Vision*, IEEE, Piscataway, NJ **2019**, pp. 6202–6211.
[16] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, L. S. Davis, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2016**, pp. 733–742.
[17] W. Luo, W. Liu, S. Gao, in *Proc. of the IEEE International Conf. on Computer Vision*, IEEE, Piscataway, NJ **2017**, pp. 341–349.
[18] Y. Zhao, B. Deng, C. Shen, Y. Liu, H. Lu, X.-S. Hua, in *Proc. of the 25th ACM International Conf. on Multimedia*, ACM, CA, USA **2017**, pp. 1933–1941.
[19] D. Gong, L. Liu, V. Le, B. Saha, M. R. Mansour, S. Venkatesh, A. V. D. Hengel, in *Proc. of the IEEE/CVF International Conf. on Computer Vision*, IEEE, Piscataway, NJ **2019**, pp. 1705–1714.
[20] H. Park, J. Noh, B. Ham, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2020**, pp. 14372–14381.
[21] H. Vu, T. D. Nguyen, T. Le, W. Luo, D. Phung, in *Proc. of the AAAI Conf. on Artificial Intelligence*, Vol. *33*, AAAI Press, Honolulu, HI, USA **2019**, pp. 5216–5223.
[22] J. T. Zhou, J. Du, H. Zhu, X. Peng, Y. Liu, R. S. M. Goh, *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 2537.
[23] W. Luo, W. Liu, D. Lian, J. Tang, L. Duan, X. Peng, S. Gao, *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *43*, 1070.
[24] T.-N. Nguyen, J. Meunier, in *Proc. of the IEEE/CVF International Conf. on Computer Vision*, IEEE, Piscataway, NJ **2019**, pp. 1273–1283.
[25] C. Park, M. Cho, M. Lee, S. Lee, in *Proc. of the IEEE/CVF Winter Conf. on Applications of Computer Vision*, IEEE, Piscataway, NJ **2022**, pp. 2249–2259.
[26] D. Xu, W. Ouyang, E. Ricci, X. Wang, N. Sebe, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2017**, pp. 5363–5371.
[27] R. T. Ionescu, F. S. Khan, M.-I. Georgescu, L. Shao, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2019**, pp. 7842–7851.
[28] T. Akilan, Q. J. Wu, A. Safaei, J. Huo, Y. Yang, *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 959.
[29] V. Dumoulin, F. Visin, *A Guide to Convolution Arithmetic for Deep Learning*, arXiv:1603.07285.
[30] C.-C. J. Kuo, *J. Vis. Commun. Image Represent.* **2016**, *41*, 406.
[31] S. Ioffe, C. Szegedy, in *International Conf. on Machine Learning*, PMLR, Lille, France **2015**, pp. 448–456.
[32] A. B. Chan, N. Vasconcelos, *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *30*, 909.
[33] A. B. Chan, N. Vasconcelos, *IEEE Trans. Image Process.* **2011**, *21*, 2160.
[34] J. Gao, L. Gan, F. Buschendorf, L. Zhang, H. Liu, P. Li, X. Dong, T. Lu, *IEEE Internet Things J.* **2020**, *8*, 951.
[35] O. Voican, *Inform. Econ.* **2021**, *25*, 70.
[36] J. Azimjonov, A. Özmen, *Adv. Eng. Inform.* **2021**, *50*, 101393.
[37] J. Yang, F. Yang, L. Zhang, R. Li, S. Jiang, G. Wang, L. Zhang, Z. Zeng, *Neurocomputing* **2021**, *444*, 170.
[38] S. S. L. Pereira, J. Maia, *Int. J. Comp. Appl.* **2021**, *183*, 1.
[39] P. Kamat, R. Sugandhi, in *E3S Web of Conf.*, Vol. *170*, EDP Sciences, Les Ulis, France **2020**, p. 02007.
[40] R. Tudor Ionescu, S. Smeureanu, B. Alexe, M. Popescu, in *Proc. of the IEEE International Conf. on Computer Vision*, IEEE, Piscataway, NJ **2017**, pp. 2895–2903.
[41] C. Sun, Y. Jia, Y. Hu, Y. Wu, in *Proc. of the 28th ACM International Conf. on Multimedia*, **2020**, pp. 184–192.
[42] W. Luo, W. Liu, S. Gao, in *2017 IEEE International Conf. on Multimedia and Expo (ICME)*, IEEE, Piscataway, NJ **2017**, pp. 439–444.
[43] Y. Qiang, S. Fei, Y. Jiao, *IEEE Access* **2021**, *9*, 68108.
[44] Y. Yang, D. Zhan, F. Yang, X.-D. Zhou, Y. Yan, Y. Wang, in *2020 IEEE 6th International Conf. on Computer and Communications (ICCC)*, IEEE, Piscataway, NJ **2020**, pp. 1832–1839.
[45] B. Ramachandra, M. Jones, in *Proc. of the IEEE/CVF Winter Conf. on Applications of Computer Vision*, IEEE, Piscataway, NJ **2020**, pp. 2569–2578.
[46] S. Zhang, M. Gong, Y. Xie, A. K. Qin, H. Li, Y. Gao, Y.-S. Ong, *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 5427.
[47] Y. Lu, K. M. Kumar, in *2019 16th IEEE International Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, IEEE, Piscataway, NJ **2019**, pp. 1–8.
[48] J. T. Zhou, L. Zhang, Z. Fang, J. Du, X. Peng, Y. Xiao, *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *30*, 4639.
[49] K. Simonyan, A. Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, arXiv:1409.1556.
[50] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2016**, pp. 2818–2826.
[51] W. Ullah, A. Ullah, I. U. Haq, K. Muhammad, M. Sajjad, S. W. Baik, *Multimed. Tools Appl.* **2021**, *80*, 16979.
[52] W. Ullah, T. Hussain, F. U. M. Ullah, M. Y. Lee, S. W. Baik, *Eng. Appl. Artif. Intell.* **2023**, *123*, 106173.
[53] S. Ul Amin, M. Ullah, M. Sajjad, F. A. Cheikh, M. Hijji, A. Hijji, K. Muhammad, *Mathematics* **2022**, *10*, 1555.