Discover Artificial Intelligence

Perspective

Exploring Convolutional Recurrent architectures for anomaly detection in videos: a comparative study

Ambareesh Ravi[1] · Fakhri Karray[1,2]

Received: 24 June 2021 / Accepted: 4 August 2021
Published online: 22 September 2021

Abstract

Convolutional Recurrent architectures are currently preferred for spatio-temporal learning tasks in videos to the 3D convolutional networks which accompany a huge computational burden and it is imperative to understand the working of different architectural configurations. But most of the current works on visual learning, especially for video anomaly detection, predominantly employ ConvLSTM networks and focus less on other possible variants of Convolutional Recurrent configurations for temporal learning which warrants a need to study the different possible variants to make informed, optimal design choices according to the nature of the application at hand. We explore a variety of Convolutional Recurrent architectures and the influence of hyper-parameters on their performance for the task of anomaly detection. Through this work, we also intend to quantify the efficiency of the architectures based on the trade-off between their performance and computational complexity. With comprehensive quantitative and visual evidence, we establish that the ConvGRU based configurations are the most effective and perform better than the popular ConvLSTM configurations on video anomaly detection tasks, in contrast to what is seen from the literature.

Keywords Video anomaly detection · ConvLSTM · ConvRNN · ConvGRU · Seq2Seq architectures

1 Introduction

Understanding videos has been one of the most challenging and open problems in computer vision [1–3] for applications such as action recognition, scene description, video captioning, video summarization and video anomaly detection. Video Anomaly Detection (VAD) is the process of identifying abnormal, rare and novel events concerning time and region of the video frames with several real-world applications in areas like security, surveillance [4–8], manufacturing [9], medicine [10] etc. Deep learning and Convolutional Neural Networks are predominantly used for visual tasks owing to their superior performance which can be attributed to their ability to uncover and learn hidden patterns and generalize well on huge datasets. But most of the prevalent deep learning architectures require heavy computational and huge memory storage resources prohibiting them from being used on edge devices for small applications and on-premise computation for data privacy reasons. In systems involving real-time detection and alerts like video surveillance, the model needs to be highly efficient in inference and accurate with decisions.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s44163-021-00004-2.

✉ Ambareesh Ravi, ambareesh.ravi@uwaterloo.ca; Fakhri Karray, karray@uwaterloo.ca | [1]Department of Electrical and Computer Engineering, Center for Pattern Analysis and Machine Intelligence (CPAMI), University of Waterloo, Ontario N2L 3G1, Canada. [2]Muhammad Ben Zayed University of AI, Masdar City, Abu Dhabi, UAE.

Videos are dynamic multi-dimensional complex data with intricate variations in spatial context over time, encompassing motion patterns of objects and entities in them. Normal events in videos exhibit definite, regular temporal patterns when compared to the portions with anomalies that exhibit contorted, aberrant patterns and learning to identify those portions will give additional robustness for applications involving temporal coherence in inputs. Although a video can be regarded as a stacked set of frames, there is a temporal coherence with the events occurring across the frames to represent motion patterns. It is vital to learn the connection between frames with the temporal correlation and is not possible to learn them with spatial models such as 2D convolution networks like a convolutional AutoEncoder that is popularly used for reconstruction-based anomaly detection. Hence, architectures like ConvLSTM that combine spatial learning from convolutional layers and temporal learning from recurrent layers are utilized and have been proven effective in tasks involving sequential modelling and understanding temporal context. In this work, we explore other variants of Convolutional Recurrent configurations such as ConvRNN and ConvGRU apart from the popular ConvLSTM, differing in the internal learning mechanisms and computational requirements. Moreover, these configurations can be employed in several Convolutional Recurrent architectures such as Convolutional Recurrent AutoEncoder (CRAE), BiDirectional Convolutional Recurrent AutoEncoder (BiCRAE) which operate on compression and reconstruction of video segments and the sequence-to-sequence Convolutional Recurrent Networks (Seq2Seq-CRN) that belong to the category of predictive models. The novel contributions from our research are (1) the use of Convolutional Recurrent layers with kernels of various sizes and strides as opposed to fixed size and unit stride layers used in most of the works[1], (2) the use of transpose Convolutional Recurrent cells with the capability of upsampling data instead of convolutional cells being used in the decoder as in most works, (3) we evaluate the effectiveness of ConvRNN and ConvGRU cells which are seldom used for video-related tasks and are not as popular compared to ConvLSTM and (4) to the best of our knowledge, this work is the first of its kind to design and jointly evaluate BiDirectional and Sequence-to-Sequence Convolutional Recurrent models for video anomaly detection. Hence, we believe that this study could prove helpful in making the right design choices for various applications that involve video understanding in an unsupervised learning setup. The key objectives of our research can be summarized as follows:

1. To obtain a qualitative understanding of the learning mechanisms of different Convolutional Recurrent configurations.
2. To analyze and quantitatively assess the true benefit of employing different Convolutional Recurrent architectures for video anomaly detection over 2D and 3D convolutional architectures.
3. Compare the effectiveness of different Convolutional Recurrent Neural Network (CRNN) variants based on the trade-off between performance enhancement and increase in complexity.

2 Literature review

Anomaly detection in visual data using deep learning can be categorized into reconstruction based, predictive and generative models [11, 12]. The simplest one of all is the usage of reconstruction based method of employing a variant of Convolutional AutoEncoder [13–19] that can learn to represent the input data in a compact form then reconstruct the data from the compact representation and the error between the inputs and the reconstructions are used as a metric to detect anomalies, where a higher reconstruction error denotes an anomaly and vice versa, under the assumption that the model is trained only on normal data. In this work, we focus on *reconstruction-based methods* that detect anomalies based on reconstruction error and *predictive auto-regressive methods* that predict the future normal frames from the past inputs and the deviation from the input frames is used as an indicator of anomaly. Some deep learning methods employ a 2D convolutional AutoEncoder for video anomaly detection on the basis that videos are made up of individual frames and have produced substantial results [6, 12, 20]. But videos are dynamic data that contain patterns of motion of objects in subsequent, coherent, temporally arranged frames as a time series. Temporal information is critical in understanding the context behind motion patterns in videos. For example, the normal scenario of a car driving along a highway suddenly going off-road is an anomaly and has to be detected. Such patterns can only be learnt with the help of temporal information and correlation as spatial[2] models [12, 21, 22] will not be able to identify

[1] Most works use multiple ConvLSTM layers with unit stride so that the outputs are of the same shape as the inputs.

[2] 2D convolutional models without temporal learning.

the behavioural pattern and change in motion as they will operate frame-wise and a car driven off-road on a farm might be identified by the model as normal. There are many use cases like surveillance, security, autonomous driving that involve temporal, dynamic data that require highly accurate models that can distinguish normal and anomalous inputs. In this work, we consider only the Convolutional Recurrent models that are capable of joint spatio-temporal learning from videos.

The work in [23] provides a comprehensive discussion on deep learning methods for anomaly detection in surveillance videos and discuss the open problems and analysis of supervised and unsupervised methods. Doshi et al. [24] proposed a two-stage method for object detection and using KNNs with optical flow features for human-in-loop anomaly detection in videos. Hasan et al. [19] used two networks, one with handcrafted features and another spatio-temporal AutoEncoder to learn the notion of regularity or normality from video data. 3D CAE is huge in terms of the number of trainable parameters and operationally inefficient when compared to the modern Convolutional Recurrent networks for tasks like action recognition. Sultani et al. [25] use a supervised method of using 3D Convolutional features with multiple instances learning to detect anomalies in real-world videos as a two stage method that lacks joint learning. The works [26, 27] have shown that 3D convolutional networks do not learn efficient representations of videos. As a result, [28] first proposed the idea of using visual features from models trained on ImageNet using transfer learning in LSTM networks to effect learning spatio-temporal features for video-related tasks like action recognition. Then, ConvLSTM which replaces fully connected layers with convolutional layers to operate on video frames was first introduced in [29] for predicting rainfall intensity patterns from the past images over a local region which was originally inspired from [30]. Later, Srivatsava et al. [31] used convolutional LSTM (ConvLSTM) networks to learn video representations in a synthetic dataset called MovingMNIST [31] which contains MNIST digits moving in definite patterns. Luo et al. [32] proposed to use ConvLSTM AutoEncoders that consists of ConvLSTM layers for the task of anomaly detection in videos by reconstructing frames from the memorized past frames and compare the results with 3D CAE on the MovingMNIST dataset. Medel et al. [33] proposed a hybrid-predictive ConvLSTM network that can both reconstruct past frames and predict future frames whereas [34] proposed to use an AutoEncoder model with three ConvLSTM layers in between 2D convolutional layers for detecting anomalous events in videos and apply Persistence 1D algorithm on regularity scores for better performance. It is established from all the previously mention-works that Convolutional Recurrent networks are effective for learning video features along with the fact that only ConvLSTMs are predominantly used. The models that are equipped to handle such spatio-temporal data are highly complex and require tremenodus resources to function. Hence, it is important to analyse different possible architectures to pick the highly efficient one with the right balance between the accuracy of predictions and amount of required computation which we intend to perform through this research.

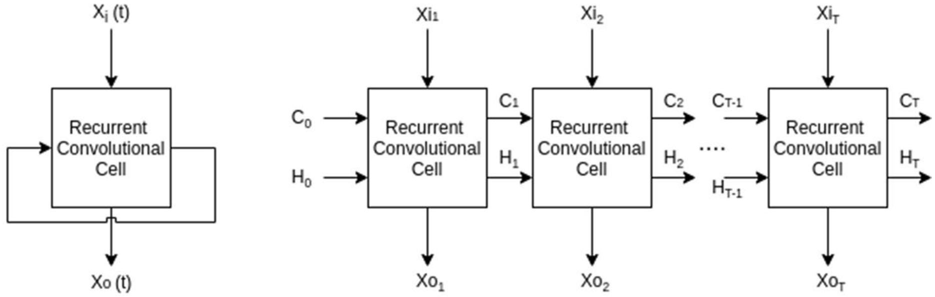
3 Methodology

The fully connected layers in Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) facilitate dense connection between the inputs and state transitions which is not optimal for learning spatial information [29] and Convolutional Recurrent architectures consists of convolutional layers instead of fully connected layers that are proven to be inherently superior for visual tasks and a natural fit for learning, abstracting and propagating spatial information thereby cogently learning spatio-temporal information from the layers due to unrolling. This section focuses on architectures comprising of such Convolutional Recurrent layers that can learn regular spatio-temporal patterns in videos for reconstructing the current or predicting the future set of frames. The primary hypothesis of the proposed solutions is that the ability of Convolutional Recurrent architectures to identify anomalies in videos should be superior to the conventional 2D Convolutional AutoEncoders.

3.1 Convolutional recurrent cells

There are two other configurations of Convolutional Recurrent cells like ConvRNN and ConvGRU apart from the popularly used ConvLSTM [29] that can be applied for the task of learning in videos. Convolutional Recurrent Cells (ConvRec Cells) are the building blocks for the Convolutional Recurrent networks (CRN). The internal connection between different time steps of the cells form a dynamic directed acyclic graph to learn input sequences over multiple time steps and this is pictorially represented in Fig. 1 where X_i, X_o denote the input and output respectively with cell state parameters

Fig. 1 Generic representation of Convolutional Recurrent layer operation



(a) Rolled representation of Recurrent Convolutional layer

(b) Unrolled representation of Recurrent Convolutional layer

like C_t, H_t which are the cell state and hidden state respectively where the suffix represents the updated time steps[3] represented by *T* indicating the number of frames (or time steps) processed. The ConvRec Cells utilize back-propagation through time (BPTT) similar to fully-connected recurrent neural networks to propagate gradients to the early time steps facilitate learning. For upsampling activations, transpose convolutions are used instead of convolutions in the decoder parts of the Convolutional Recurrent architectures. The blocks marked with *2D Conv* are made up of a convolutional (or transpose convolutional layer), a batch normalization layer and a Rectified Linear Unit (ReLU) layer in tandem. This section discusses the learning mechanisms of different ConvRec cells in detail as it is essential to understand the working of different architectures. The representation of the different ConvRec cells are available in Additional file 1.

3.1.1 ConvRNN cell

The ConvRNN cell adopts the structure of vanilla Recurrent Neural Networks (RNNs) [35] with convolutional layers instead of fully connected layers to effect sequential learning in video data. The ConvRNN cel consists of a hidden state and output state with each state accompanied with designated weights. The current hidden state is a function of the previous hidden state and the current input and is passed on to the next time step. The current output state is a function of the activated current hidden state. The inputs to Convolutional Recurrent architectures are of the shape $B \times T \times W \times H \times C$ where *B* is the batch size, *T* the number of time steps/recurrent unrolling or number of frames in the video clip, $W \times H \times$ being the frame size with width, height and channels respectively and only the batch of single frame per time step of shape $B \times W \times H \times C$ is passed to the recurrent network enabling the re-usability of weights across time steps to persist information that is required in the learnt states i.e. *memory*. The equations governing the operation of ConvRNN cell are shown in 1 where $*$ represents the convolution operation, *W*, *b* the weights and biases, *X*, *H*, *O* the input, hidden and output states respectively at the time step *t*, σ the Sigmoid activation function.

$$
\begin{aligned}
H_t &= \tanh(W_{xi} * X_t + W_{hi} * H_{t-1} + b_i) \\
O_t &= \sigma(W_{ho} * H_t + b_o)
\end{aligned}
\tag{1}
$$

3.1.2 ConvLSTM cell

LSTM introduced in [36], has achieved significant performance improvements on sequence modelling, language modelling and other natural language processing tasks over the vanilla RNN models. The major improvement in LSTM over RNN is the ability to avoid vanishing, exploding gradients and maintaining a *cell state* to learn and retain long term dependencies better. An ConvLSTM cell consists of an input gate *i*, output gate *o*, forget gate *f* and a cell state *C*. The three gates regulate the information pass in and out of the cell using convolution operations whereas the cell state persists information in memory over long periods. The Eq. 2 shows the operation of ConvLSTM cell for inputs as discussed in the previous section. The input gate propagates important information from the input frames into the other parts of the cell and the forget state helps moderation of essential information into the cell state and the cell state ultimately acts as a refined

[3] The states are initialized randomly.

memory unit that is shared across the time steps. The output state is the function of the previous hidden state, updated cell state and the input from which the new hidden state is calculated which contain the imminent information propagated from the previous time step. The $\odot$ in the equation represent Hadamard product or element-wise multiplication.

$$\begin{aligned} i &= \sigma(W_{xi} * X_t + W_{hi} * H_{t-1} + W_{ci} \odot C_{t-1} + b_i) \\ f_t &= \sigma(W_{xf} * X_t + W_{hf} * H_{t-1} + W_{cf} \odot C_{t-1} + b_f) \\ C_t &= f_t \odot C_{t-1} + i_t \odot \tanh(W_{xc} * X_t + W_{hc} * H_{t-1} + b_c) \\ o_t &= \sigma(W_{xo} * X_t + W_{ho} * H_{t-1} + W_{co} \odot C_t + b_o) \\ H_t &= o_t \odot \tanh(C_t) \end{aligned} \tag{2}$$

3.1.3 ConvGRU cell

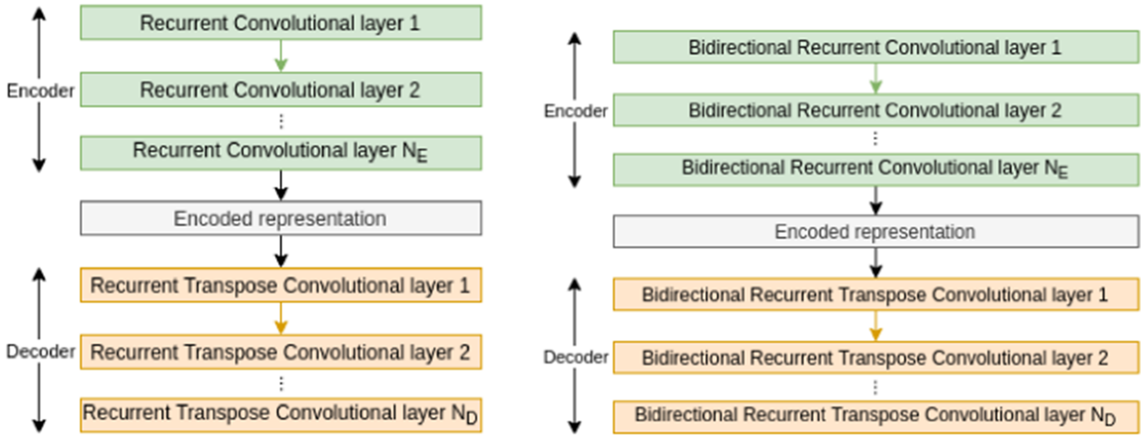
Gated Recurrent Units (GRU) [37] employ gating mechanisms in RNN and achieve better performance in some tasks like speech and music modelling in comparison with LSTM with fewer parameters on datasets with small sequences. ConvGRU with convolutional layers consists of reset *r* and updates *u* gates to regulate information flow inside the cell through an activation layer *a*. The activation layer is a function of the previous hidden state and the current updated input and the hidden layer is the transformed activated state that is used for the next time step and as an output. The states in GRU are a simplified version of the ones in LSTM. The reset gate controls the memory of the previous state that is required for reconstruction or prediction of the next frame and the update gate controls how much of the input is to be retained and the hidden state is the function of both along with the previous hidden state.

$$\begin{aligned} u &= \sigma(W_{xu} * X_t + W_{hu} * H_{t-1} + b_u) \\ r &= \sigma(W_{xr} * X_t + W_{hr} * H_{t-1} + b_r) \\ a &= \tanh(r \odot (W_{ha} * H_{t-1}) + W_{xa} * X_t) \\ H_t &= (a \odot (1 - u)) + (u \odot H_{t-1}) \end{aligned} \tag{3}$$

3.2 Convolutional Recurrent AutoEncoders

Using the Convolutional Recurrent cells discussed in the earlier sections as building blocks for learning spatio-temporal correlation from video data, it is viable to construct Convolutional Recurrent AutoEncoders with multiple layers that can encode video segments into a learnt, compressed representation and reconstruct the video from the representations with the motion under context. Three variants of Convolutional Recurrent AutoEncoders are presented in this work which are ConvRNN CAE, ConvGRU CAE, ConvLSTM CAE which share a common structure except for their respective variant of Convolutional Recurrent cells. A generic Convolutional Recurrent AutoEncoder consists of an encoder and a decoder network with Convolutional Recurrent layers and recurrent transpose-convolutional layers respectively as shown in Fig. 2a. An encoder consists of stacked layers that learn and abstract spatial dimensions into an encoder representation similar to a conventional Convolutional AutoEncoder (CAE) and a decoder upsamples the representation into rich activation maps to finally reconstruct data similar to the input video clip. Mean squared error (MSE) is used as the objective function for minimization and all other operation is similar to that of CAE except the input having an extra-temporal dimension, time steps (*T*) in the input (Fig. 2).

Each variant of Convolutional Recurrent AutoEncoder considered for experimentation consists of 5 layers of encoder and decoder. The number of kernels in the encoder is 64, 64, 64, 96, 96 and the sizes of the kernels are 3×3 with stride 2 except for the last layer of the encoder and the first layer of a decoder which have stride as 1. The decoder is the mirror equivalent of the encoder. In both encoder and decoder, L_R layers closer to the bottleneck are of recurrent type and the remaining $L - L_R$ ($L = 5$ in our case) are time distributed 2D convolutional or transpose convolutional layers. Each convolutional and transpose convolutional block contain batch normalization and Leaky ReLU activation and the final layer of decoder has Sigmoid activation.

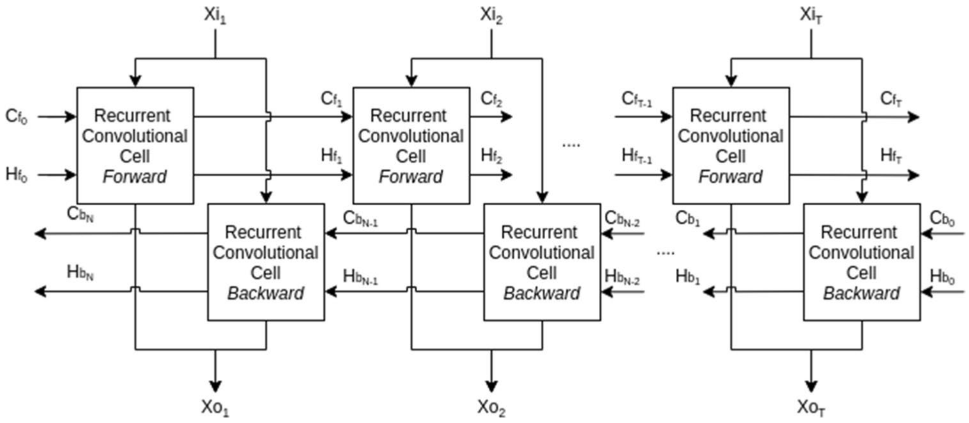

Encoder
Recurrent Convolutional layer 1
Recurrent Convolutional layer 2
Recurrent Convolutional layer N_E
Encoded representation
Decoder
Recurrent Transpose Convolutional layer 1
Recurrent Transpose Convolutional layer 2
Recurrent Transpose Convolutional layer N_D
Encoder
Bidirectional Recurrent Convolutional layer 1
Bidirectional Recurrent Convolutional layer 2
Bidirectional Recurrent Convolutional layer N_E
Encoded representation
Decoder
Bidirectional Recurrent Transpose Convolutional layer 1
Bidirectional Recurrent Transpose Convolutional layer 2
Bidirectional Recurrent Transpose Convolutional layer N_D

(a) Structure of Convolutional Recurrent AutoEncoder

(b) Structure of BiDirectional Convolutional recurrent AutoEncoder

Fig. 2 General representation of Convolutional Recurrent AutoEncoder architectures

Fig. 3 Representation of a generic BiDirectional Convolutional Recurrent layer



3.3 Bidirectional Convolutional Recurrent AutoEncoders

Similar to the Convolutional Recurrent AutoEncoder discussed in the previous section, a bidirectional Convolutional Recurrent AutoEncoder (Fig. 2b) has the same architecture except that the Convolutional Recurrent layers are bidirectional that can learn from the ordered and temporally reversed inputs under the intuition that the AutoEncoder can learn both from past and future input time steps and have proven advantages and performance enhancement in tasks involving understanding context from data, especially predictive tasks. To the best of the our knowledge, we are the first to design and evaluate bidirectional variants for Convolutional Recurrent architectures for anomaly detection task. The bidirectional Convolutional Recurrent cell consists of two modules, a *forward* and *backward* module each equivalent to a vanilla Convolutional Recurrent cell. The forward module operates normally as stated in the previous section and the backward module operates by learning information from the temporally-reversed input data batch as shown in Fig. 3. Finally, the output from the two modules is combined to produce the final five-dimensional activation maps and passed on to the next layer. The forward and backward outputs are aggregated by taking average in the temporal dimension.[4] Three variants are used for experiments—BiDirectional ConvRNN AutoEncoder, BiDirectional ConvGRU AutoEncoder and BiDirectional ConvLSTM AutoEncoder.

[4] There are several methods to combine the outputs from the forward and backward modules like addition, concatenation, multiplication, dot product etc.

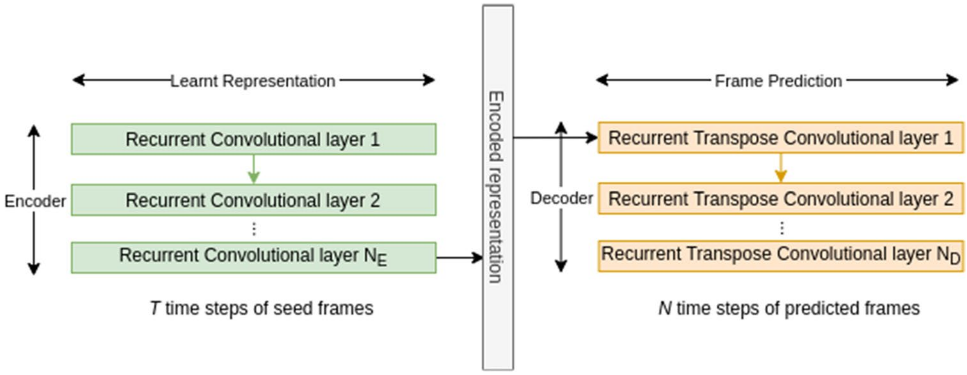

Learnt Representation
Recurrent Convolutional layer 1
Recurrent Convolutional layer 2
Recurrent Convolutional layer N_E
Encoder
T time steps of seed frames
Encoded representation
Frame Prediction
Recurrent Transpose Convolutional layer 1
Recurrent Transpose Convolutional layer 2
Recurrent Transpose Convolutional layer N_D
Decoder
N time steps of predicted frames

Fig. 4 Structure of Seq2Seq Convolutional Recurrent architecture

3.4 Sequence to sequence Convolutional Recurrent models

Seq2Seq models are a special variant of recurrent architectures belonging to the category of *auto-regressive* models that are used for modelling time-series data to learn sequence from domain one and to transform the learnt knowledge into prediction in the same or different domain and are widely used in Natural Language Processing (NLP) tasks. The goal of Seq2Seq models for anomaly detection is to learn normalcy and predict the future frames from a set of seed input frames as opposed to mere reconstruction as in the previously discussed models. The hypothesis is that the normal patterns of motion in videos learnt while training can be easily predicted similar to *cause and effect* phenomenon and the model will be able to predict the future of normal events with a high degree of certainty almost matching the rest of the input video clip. Seq2seq models are seldom employed and evaluated in the literature for anomaly detection, we consider this experiment to be an important contribution of our work. This architecture is trained with sets of input seed frames ($N_{seed} = 4$) and the error between the rest of the input frames and the predicted frames ($N_{pred} = 4$) is minimized using MSE as the objective function. Eventually, a well-trained model on normal data will fail to predict the future of an initiated anomalous event. This comparison between an actual and predicted set of frames helps in the quantification of anomalies. A Convolutional Recurrent AutoEncoder with an encoder and a decoder can be re-purposed into a Seq2Seq model where the major difference is in the inputs and the overall learning mechanisms as the latter use the states and embedding from the last time step for predicting the future frames as represented in Fig. 4 as opposed to features at all time steps in CRAE. For the experiments, three variants of Seq2Seq architecture—Seq2Seq ConvRNN CAE, Seq2Seq ConvGRU CAE and Seq2Seq ConvLSTM CAE are used.

4 Experiments and results

The focus of this work is to explore the efficacy of different varieties of Convolutional Recurrent networks for video anomaly detection performance and not to create new models. In the experiments two important parameters are varied to study their effects—the number of recurrent layers in encoder & decoder L_R and the option to replace recurrent deconvolutional layers ($DUT = N$) with *Time Distributed* spatial (2D convolutional) layers ($DUT = Y$) in the *decoder*. The explanations of the hyper-parameters are provided in Table 2. The former is important to study the effectiveness and role of recurrent layers in learning patterns in early stages before abstraction and the latter is to understand if spatial (transpose) convolutions can reconstruct or predict frames as good as recurrent layers from the latent embeddings that contain temporal information from the early recurrent layers in the encoder. For all the architectural variants, the learning in the model is conditioned in such a way that the important information is contained out of the ultimate encoder layer which is vital for the reconstruction of existing or predicting the future frames. To test the spatio-temporal learning in the models, the datasets are chosen in such a way that they contain both spatial and temporal anomalies. For example, Avenue dataset contains spatial anomalies like a bag on the floor and temporal anomalies like people (normal entities in frames) jumping. For evaluating the performance of models, we use popular metrics for anomaly detection such as Area under Receiver Operating Characteristics score (AUC-ROC score) from the plot between False Positive Rate and True

Table 1 Details of the anomaly detection datasets

Dataset	Train	Test	Anomalies
CUHK avenue [38]	16 videos	21 videos	Run, jump, throw, catch, walk, objects
UCSD Ped 1 [39]	34 videos	16 videos	Bike, cart, walking on grass, skateboards
UCSD Ped 2 [39]	36 videos	12 videos	
Subway entrance [40]	15 min	5 min	Skipping payments, loitering, jumping turnstiles
Subway exit [40]	5 min	40 min	

Positive Rate which denotes the ability of a model to distinguish normal samples from the abnormal ones, and Equal Error Rate (EER) is a point on the ROC curve where the false positive and negative rates are equal[5]. We also use other common metrics such as precision, recall and F1-Score.

4.1 Datasets

To enunciate the performance of our proposed approaches, we consider 5 video datasets for the experiments on the proposed models. The frequency of anomalies vary from video to video in each of the datasets. Anomalies in these datasets are mostly contextual,[6] based on objects in the frame and their motion patterns. The detailed information of the nature of anomalies in each datasets are presented in Table 1. The statistics of the datasets are provided in Table 1 and are described briefly in this section. CUHK Avenue dataset [38] consists of 2 minutes long videos with frame-level ground truth. Anomalies occur both in the background and foreground and the training set consists of a few unrecorded anomalies too. UCSD pedestrian datasets [39] 1 and 2 deal with abnormal events in pedestrian motion. Both Ped 1 and Ped 2 have frame-wise temporal annotations. The subway datasets [40] depict a surveillance scenario with two cameras in a subway station. The dataset consists of event-level ground truth and hence, based on manual inspection, we use a window of 15 frames on either side of the temporal label to replicate the labels although some events seem to last longer up to 50 frames in the subway datasets.

4.2 Experimental setup

The experiments are conducted on 9 different architectures—ConvRNN AutoEncoder (CRNN AE), ConvLSTM AutoEncoder (CLSTM AE), ConvGRU AutoEncoder (CGRU AE), BiDirectional ConvRNN AutoEncoder (BiCRNN AE), BiDirectional ConvLSTM AutoEncoder (BiCLSTM AE), BiDirectional ConvGRU AutoEncoder (BiCGRU AE), Seq2Seq ConvRNN network (Seq2Seq CRNN NN), Seq2Seq ConvLSTM network (Seq2Seq CLSTM NN), Seq2Seq ConvGRU network (Seq2Seq CGRU NN) with variation in two important parameters—L_R and *DUT* on 5 different video datasets. The inputs frames are resized to 128×128 and are arranged as tensors of shape $T \times W \times H \times C$. The normal frames are labelled as 1 and anomalies as 0. The models are trained only on normal data for 300 epochs using MSE as objective function and Adam optimizer with a starting learning rate of 1×10^{-03} equipped with learning rate decay and early stopping with a batch size of 32 in a computing cluster. The dataset is augmented with varying strides of frames such as 1, 2, 4, 8, 16 prior to training and the test set is retained as such without any change. The error/loss are calculated between every pair of input and predicted frames and are used for performance evaluation. For each of the models, the frame-wise losses are calculated using MSE and temporally aggregated per video. The aggregated loss $e(t)$ at time t are used to calculate the regularity $s(t)$ which denotes the probability of a frame being normal[7]. The temporal regularity per video $s(t)$ is calculated using the Eq. 4 where $I(x, y, t)$ is the pixel intensity at position x, y at time step t. *Sav-Gol filter* is applied on the regularity for a window of 15 frames instead of the Persistence 1D [41] algorithm on a window of 50 frames used by many works. This process helps smoothing local minima or maxima. For the experiments[8], we use PyTorch and testing was carried out on a computer with Intel Core i7-6700K, 32 GB RAM and NVIDIA GeForce GTX 1070 8GB VRAM (Table 2).

[5] AUC-ROC—higher the better, an ideal model has value 1.0 and a poor model has 0.0; EER—lower the better.

[6] The study does not consider anomalies due to data transmission or streaming loss.

[7] Regularity is 1.0 for a perfectly normal frame and lower for anomalous frame.

[8] Complete codebase available at https://github.com/ambareeshravi/Thesis_VideoAnomalyDetection/.

Table 2 Configurable hyper-parameters in Convolutional Recurrent models

Parameter	Description	Effect studied?
Image size $W \times H$	The performance of models increase with increase in resolution and saturate after a maximum resolution $W_{max} \times H_{max}$. Directly proportional to computational requirement	Fixed to 128×128
Channels *C*	Grayscale images have 1 channel and colored images have 3 channels	Depends on the input
Time steps *T*	Time steps or number of frames per unrolling/video clip, as rightly discussed by [19] has little to no effect on performance	Fixed to 8
Number of layers *L*	Total number of layers in encoder or decoder signifies how deep an architecture is	Fixed to 5
Number of recurrent layers L_R	The remaining $L - L_R$ layers are time distributed 2D convolutional layers	Varied from 1 to 3
Decoder upsampling type *DUT*	Decoder upsampling type represents the nature of the decoder whether recurrent transpose convolutional layers are used or only time distributed 2D transpose convolutions are used instead	Both variants studied[a]

[a] *N* if recurrent upsampling and *Y* if 2D transpose convolutional upsampling i.e. if Convolutional Recurrent layers are disabled

Table 3 Performance comparison of models on different datasets [variants represented by (L_p, *DUT*)]

Model	Avenue		Subway entrance		Subway exit		UCSD1		UCSD2	
	AUC ROC	EER	AUC ROC	EER	AUC ROC	EER	AUC ROC	EER	AUC ROC	EER
CRNN(1,N)	0.83	0.24	0.78	0.31	0.96	0.11	0.69	0.35	0.82	0.24
CRNN(1,Y)	0.83	0.25	0.76	0.33	0.96	0.11	0.69	0.36	0.83	0.26
CRNN(2,N)	0.71	0.35	0.79	0.27	0.96	0.11	0.73	0.32	0.83	0.22
CRNN(2,Y)	0.83	0.23	0.76	0.31	0.96	0.11	0.68	0.36	0.79	0.28
CRNN(3,N)	0.68	0.36	0.78	0.28	0.96	0.11	0.67	0.36	0.70	0.31
CRNN(3,Y)	0.71	0.35	0.79	0.27	0.96	0.11	0.73	0.33	0.85	0.23
CLSTM(1,N)	0.82	0.23	0.74	0.34	0.96	0.11	0.68	0.36	0.77	0.27
CLSTM(1,Y)	0.78	0.27	0.74	0.34	0.96	0.11	0.65	0.39	0.86	0.22
CLSTM(2,N)	0.83	0.24	0.68	0.37	0.95	0.11	0.70	0.35	0.85	0.22
CLSTM(2,Y)	0.78	0.26	0.74	0.32	0.96	0.11	0.68	0.35	0.80	0.30
CLSTM(3,N)	0.82	0.25	0.75	0.34	0.96	0.11	0.69	0.34	0.84	0.23
CLSTM(3,Y)	0.83	0.25	0.71	0.35	0.95	0.11	0.69	0.35	0.81	0.28
CGRU(1,N)	0.80	0.24	0.74	0.33	0.95	0.11	0.67	0.36	0.81	0.25
CGRU(1,Y)	0.85	0.25	0.75	0.33	0.96	0.11	0.67	0.37	0.86	0.21
CGRU(2,N)	0.78	0.25	0.69	0.38	0.96	0.11	0.70	0.35	0.86	0.22
CGRU(2,Y)	0.78	0.27	0.71	0.34	0.96	0.11	0.67	0.36	0.83	0.23
CGRU(3,N)	0.82	0.27	0.75	0.32	0.96	0.11	0.69	0.35	0.85	0.23
CGRU(3,Y)	0.81	0.24	0.70	0.36	0.96	0.11	0.66	0.37	0.58	0.44
BiCRNN(1,N)	0.78	0.26	0.77	0.31	0.96	0.11	0.68	0.36	0.80	0.29
BiCRNN(1,Y)	0.84	0.26	0.76	0.31	0.96	0.11	0.67	0.38	0.81	0.27
BiCRNN(2,N)	0.70	0.36	0.79	0.27	0.96	0.11	0.73	0.33	0.78	0.32
BiCRNN(2,Y)	0.83	0.23	0.76	0.31	0.96	0.11	0.68	0.37	0.84	0.23
BiCRNN(3,N)	0.65	0.41	0.79	0.27	0.96	0.11	0.67	0.36	0.75	0.29
BiCRNN(3,Y)	0.71	0.35	0.79	0.27	0.96	0.11	0.73	0.33	0.85	0.23
BiCLSTM(1,N)	0.78	0.26	0.73	0.33	0.96	0.11	0.68	0.36	0.85	0.24
BiCLSTM(1,Y)	0.79	0.27	0.74	0.33	0.96	0.11	0.68	0.38	0.82	0.26
BiCLSTM(2,N)	0.76	0.29	0.62	0.42	0.96	0.11	0.69	0.36	0.70	0.31
BiCLSTM(2,Y)	0.78	0.26	0.77	0.32	0.96	0.11	0.68	0.36	0.85	0.23
BiCLSTM(3,N)	0.79	0.27	0.76	0.33	0.97	0.11	0.70	0.34	0.85	0.22
BiCLSTM(3,Y)	0.78	0.28	0.74	0.33	0.95	0.11	0.70	0.35	0.85	0.23
BiCGRU(1,N)	0.76	0.29	0.72	0.33	0.96	0.11	0.68	0.36	0.81	0.27
BiCGRU(1,Y)	0.78	0.26	0.74	0.34	0.96	0.11	0.68	0.37	0.84	0.25
BiCGRU(2,N)	0.75	0.29	0.59	0.45	0.96	0.11	0.69	0.35	0.85	0.21
BiCGRU(2,Y)	0.73	0.30	0.75	0.32	0.96	0.11	0.66	0.38	0.79	0.24
BiCGRU(3,N)	0.79	0.27	0.74	0.32	0.97	0.11	0.69	0.34	0.76	0.28
BiCGRU(3,Y)	0.75	0.27	0.69	0.35	0.96	0.11	0.68	0.36	0.85	0.23
Seq2Seq CRNN(1,N)	0.71	0.35	0.85	0.23	0.97	0.11	0.72	0.33	0.78	0.29
Seq2Seq CRNN(2,N)	0.70	0.36	0.79	0.27	0.96	0.11	0.73	0.33	0.82	0.27
Seq2Seq CRNN(3,N)	0.67	0.36	0.76	0.30	0.96	0.11	0.72	0.34	0.57	0.45
Seq2Seq CLSTM(1,N)	0.68	0.33	0.85	0.23	0.97	0.07	0.71	0.34	0.76	0.33
Seq2Seq CLSTM(2,N)	0.69	0.33	0.85	0.22	0.98	0.06	0.74	0.32	0.83	0.25
Seq2Seq CLSTM(3,N)	0.67	0.35	0.86	0.22	0.98	0.07	0.74	0.31	0.82	0.24
Seq2Seq CGRU(1,N)	0.68	0.34	0.85	0.22	0.97	0.07	0.73	0.33	0.84	0.24
Seq2Seq CGRU(2,N)	0.69	0.34	0.85	0.22	0.97	0.07	0.74	0.32	0.86	0.20
Seq2Seq CGRU(3,N)	0.70	0.34	0.85	0.23	0.97	0.08	0.74	0.32	0.80	0.31

Table 4 Comparing Convolutional Recurrent models represented by (L_R, *DUT*) with 2D convolutional AutoEncoders and models from other works

Dataset	Model	AUC-ROC	EER	Precision	Recall	F1-score
Avenue	2D CAE	0.816	0.259	0.752	0.743	0.744
	Hand crafted + spatio-temporal CAE [19]	0.702	0.251	N/A	N/A	N/A
	ConvLSTM [12]	0.840	N/A	N/A	N/A	N/A
	Predictive ConvLSTM 224 [33]	N/A	N/A	0.952	1.00	N/A
	CGRU AE (1, Y) [***ours***]	0.853	0.249	0.804	0.796	0.792
UCSD Ped 1	2D CAE	0.683	0.375	0.672	0.672	0.662
	Hand crafted + spatio-temporal CAE [19]	0.810	0.279	N/A	N/A	N/A
	ConvLSTM [12]	0.670	N/A	N/A	N/A	N/A
	Predictive ConvLSTM 224 [33]	N/A	N/A	0.864	0.923	N/A
	Seq2Seq CLSTM NN (3,N) [***ours***]	0.737	0.310	0.696	0.694	0.694
UCSD Ped 2	2D CAE	0.838	0.265	0.849	0.669	0.7074
	Hand crafted + spatio-temporal CAE [19]	0.900	0.217	N/A	N/A	N/A
	ConvLSTM [12]	0.770	N/A	N/A	N/A	N/A
	Predictive ConvLSTM 224 [33]	N/A	N/A	0.923	1.00	N/A
	Seq2Seq CGRU NN (2,N) [***ours***]	0.862	0.192	0.875	0.790	0.813
Subway entrance	2D CAE	0.7428	0.333	0.963	0.622	0.734
	Hand crafted + spatio-temporal CAE [19]	0.940	0.260	N/A	N/A	N/A
	Predictive ConvLSTM 224 [33]	N/A	N/A	0.816	0.939	N/A
	Seq2Seq CGRU NN (3,N) [***ours***]	0.854	0.226	0.973	0.701	0.800
Subway exit	2D CAE	0.954	0.111	0.991	0.928	0.962
	Hand crafted + spatio-temporal CAE [19]	0.807	0.099	N/A	N/A	N/A
	Predictive ConvLSTM 224 [33]	N/A	N/A	0.659	0.967	N/A
	Seq2Seq CLSTM NN (2,N) [***ours***]	0.978	0.060	0.999	0.940	0.969

Bold values indicates differentiate our proposed solutions from the rest

$$\begin{aligned} e(t) &= ||I_{(x,y,t)} - f_d(f_e(I_{(x,y,t)}))||_2 \\ s(t) &= 1 - \left[\frac{(e(t) - min_{e(t)})}{(max_{e(t)} - min_{e(t)})} \right] \end{aligned} \tag{4}$$

4.3 Results and analysis

The experimental results on 9 architectures are expressed in Table 3 although a more comprehensive tabulation of performance with additional evaluation metrics such as Precision, Recall, F1-Score are available in Additional file 1. We present the findings and results of our research under various factors and contexts in this section.

4.3.1 General comparison with 2D convolutional models and models from other works

As discussed earlier, the 2D convolutional AutoEncoders (CAE) that are used for image level anomaly detection are not capable of learning temporal information from video data and hence we compare the results of our Convolutional Recurrent models with that of a baseline CAE. The CAE under consideration has exactly the same structure of the recurrent architectures but only with 2D convolutional layers operating on individual frames of the input video. The performance enhancement owing to temporal correlation from data is conspicuous from the results tabulated in Table 4, demonstrating the efficacy of using Convolutional Recurrent layers for representational learning of motion patterns in videos and

Table 5 Complexity of the various model configurations

Model	Number of trainable parameters					
	DUT = 2D TransposeConv upsampling			DUT = Recurrent upsampling		
	$L_R = 1$	$L_R = 2$	$L_R = 3$	$L_R = 1$	$L_R = 2$	$L_R = 3$
CRNN AE	0.60 M	0.76 M	0.84 M	0.76 M	1.00 M	1.15 M
CLSTM AE	1.01 M	1.51 M	1.77 M	1.59 M	2.40 M	2.92 M
CGRU AE	0.85 M	1.20 M	1.39 M	1.34 M	1.96 M	2.36 M
BiCRNN AE	0.85 M	1.23 M	1.42 M	1.26 M	1.85 M	2.22 M
BiCLSTM AE	1.67 M	2.72 M	3.28 M	2.92 M	4.65 M	5.76 M
BiCGRU AE	1.34 M	2.12 M	2.52 M	2.42 M	3.77 M	4.65 M
Seq2Seq CRNN NN	0.74 M	0.91 M	0.98 M	0.91 M	1.15 M	1.29 M
Seq2Seq CLSTM NN	1.16 M	1.65 M	1.91 M	1.74 M	2.55 M	3.06 M
Seq2Seq CGRU NN	0.99 M	1.35 M	1.53 M	1.49 M	2.11 M	2.51 M

also proving the hypothesis that temporal information is crucial in understanding the notion of normality in videos. The table also shows the results from other works which employ more complex models at a higher input resolution of 224 than the models in this work which have 128[9] and our models perform considerably well, outperforming some methods despite a smaller architectural size.

4.3.2 Performance variation due to L_R and *DUT*

The number of recurrent layers L_R has a direct effect on the overall performance of the Convolutional Recurrent architectures. The performance steadily increases with increasing L_R when $DUT = N$ i.e. when a mix of Convolutional Recurrent and time distributed convolutional layers are used in both encoder and decoder. But in the absence of Convolutional Recurrent layers in the decoder i.e. $DUT = Y$, the performance improves till $L_R = 2$ and then saturates or dips for $L_R > 2$. This shows the effectiveness of convolutional and transpose Convolutional Recurrent layers in abstracting and upsampling spatio-temporal data well. Also, the results pertaining to the change in *DUT* shows that recurrent transpose convolutional layers are superior to time distributed 2D layers in reconstructing or predicting frames in the decoder from the latent representation for detecting anomalies. Hence, the performance of architectures with at least one recurrent layer i.e. $L_R \geq 1$ in the decoder with Decoder Upsampling type as recurrent ($DUT = N$) is better than the architectures devoid of recurrent layers in the decoder ($DUT = Y$).

4.3.3 Comparison of the Convolutional Recurrent cells—CRNN vs CLSTM vs CGRU

The CRNN model variants severely suffer from memorizing the background on Avenue and UCSD1 datasets when $L_R > 1$. This can be attributed to the simpler internal mechanism that are clearly insufficient to learn the notion of normality from the variations in the input video clips. CGRU model variants consistently perform the best on all the datasets although their performance is slightly sub-par on Avenue in comparison to CLSTM variants and this scenario is interesting since Avenue is the only dataset with coloured input frames among the lot. The performance of CRNN architectures with $L_R = 1$ is considerably good, immaterial of the type of the upsampling layer.

4.3.4 Comparison of the architectural variant—normal vs BiDirectional vs Seq2Seq

Contrary to the effectiveness of BiDirectional recurrent layers in natural language tasks, their effectiveness is sub-par in video anomaly detection as they accompany a large number of parameters without a significant boost in overall performance even though they have better reconstructions and overall recall on all the datasets. This trend confirms the hypothesis that bidirectional variants possess better capability at learning and representing videos but this has an adverse effect on anomaly detection performance as they have the tendency to reconstruct anomalies too and hence

[9] separated by horizontal lines.

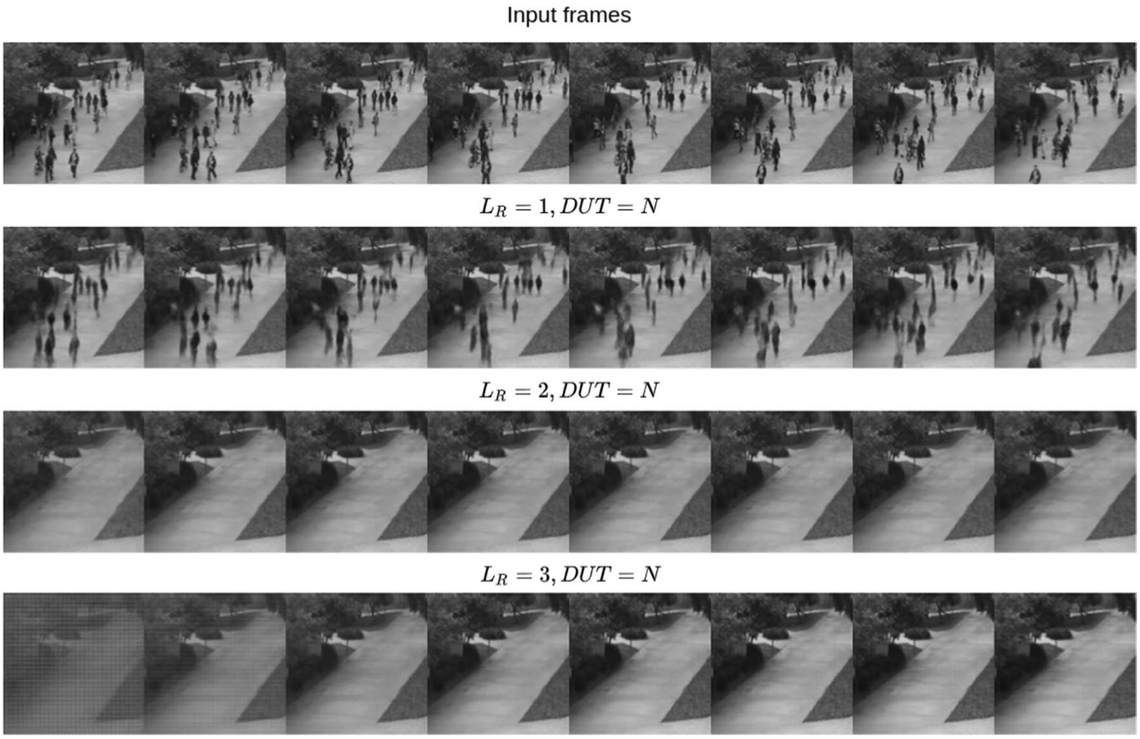


Fig. 5 Poor quality of predicted frames in Seq2Seq CLSTM NN with $L_R = 2, DUT = N$

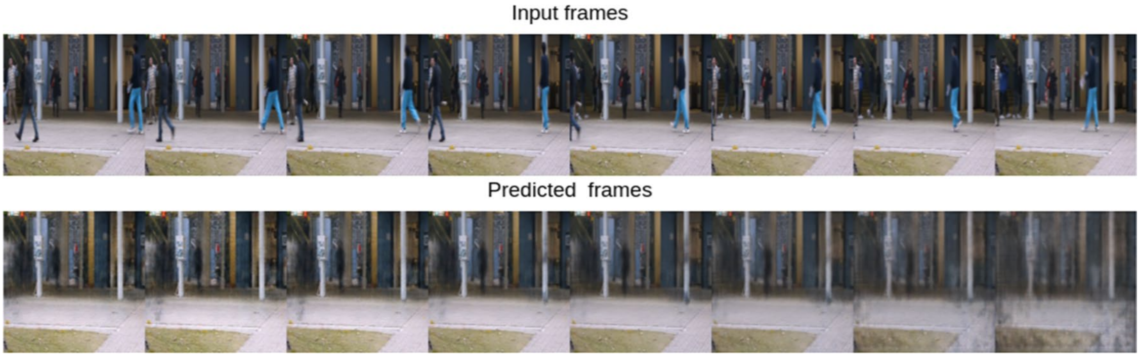


Fig. 6 CRNN variants learning the background with increasing L_R

have lower losses for anomalies, affecting their overall performance. The *Seq2Seq variants perform the best and are better-suited for anomaly detection* compared to their normal counterparts. This is due to fact that models are conditioned to predict future normality from the compact representations from the past frames through the encoding process and this is capitalized for anomaly detection as significant variations in losses are observed between predicted normal frames and input anomalous frames. The normal CRAE variants perform considerably well especially compared to 2D convolutional AutoEncoders.

4.3.5 Trade-off between performance and computational complexity

Many applications require computationally efficient models to run from edge or on-premise computing devices that are devoid of huge computation power with GPUs or TPUs. Hence, it is important in analysing the computation and inference of each of the model variants. Table 5 shows the comparison of the number of trainable parameters in each architectural configuration under consideration for coloured inputs. The use of Seq2Seq variant accompanies an increase in the number of trainable parameters by 16% on an average in comparison to the normal counterparts

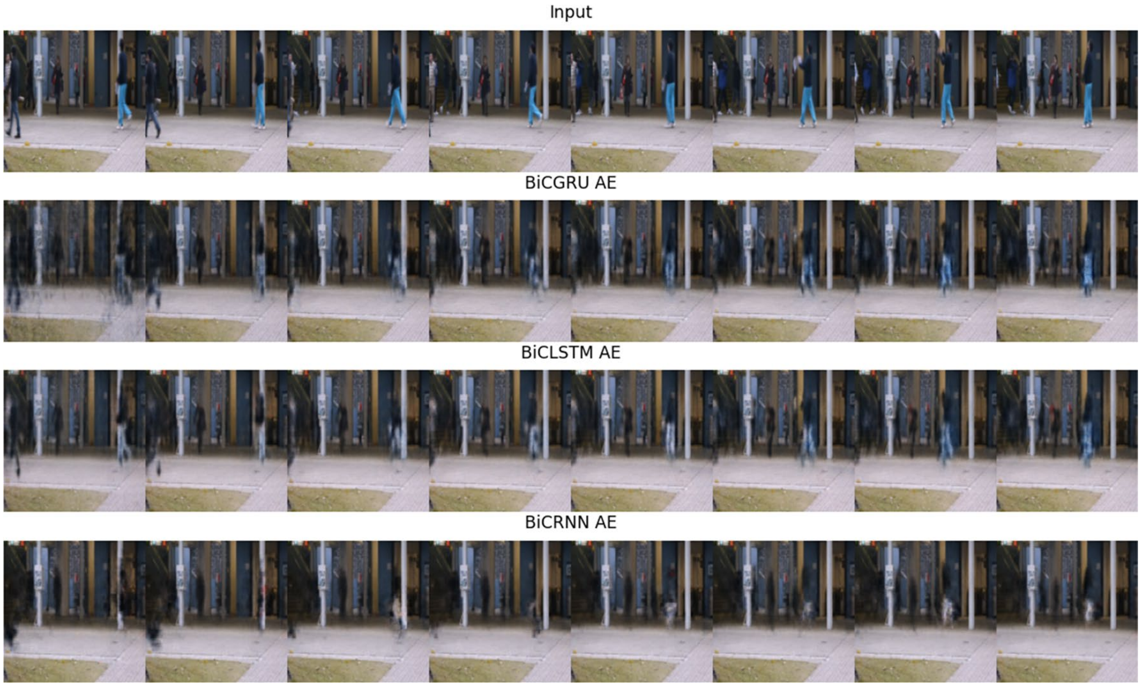


Fig. 7 BiDirectional exhibiting better learning of motion patterns

with a significant boost in anomaly detection performance. While considering the overall performance and the number of parameters that directly affect the inference and overall execution times along with the requirement of computational resources, ***Seq2Seq CGRU*** models have the right balance between performance and computational complexity and is the *overall best performing architecture* in this study for video anomaly detection. This result can be observed across the five datasets and astonishingly, almost all the works in the literature employ ConvLSTM in general for video-related tasks. The right configuration of Seq2Seq can significantly enhance the performance over simple Convolutional Recurrent AutoEncoder models and it can be attributed to the nature of learning, since the former learns the predict the motion and future events based on the learnt past sequential frames whereas the latter only performs compression and reconstruction. Finally, based on the performance metrics on the evaluated datasets in comparison to other models, ConvGRU cell is the most effective learning configuration which is in contradiction to what is seen from other popular works in the area.

4.3.6 Visual analysis of reconstructions and predictions

The reconstructions and predictions from the models have a direct impact on the overall anomaly detection performance and hence are analysed in this section. The outputs of all the model configurations for various normal and anomalous inputs from different datasets are presented in Additional file 1. The number of seed frames N_{seed} and predicted frames N_{pred} are both set to 4 for training and testing. But for the sake of analysis, the number of predicted frames is increased to 8 and the results are presented. Since the Seq2Seq models are trained to predict only up to 4 time steps, naturally the predicted frames after the 6th predicted time step are deformed and blurred as seen in Fig. 5, although one would expect the recurrent networks to perform better for longer periods. For $L_R > 1$, CRNN model variants seem to memorize the background without any useful reconstructions for both normal and abnormal data as seen in Fig. 6. This phenomenon is observed on almost all the datasets. The outputs from CLSTM AE, CGRU AE are slightly better with an increasing value of L_R. As hypothesised, BiDirectional variants exhibited reconstructions with better motion patterns but with the ability to reconstruct even the anomalous objects in the frame as seen in Fig. 7, showing capability of learning videos although this might not be suitable for anomaly detection. Moreover, the outputs of all the model variants are better when recurrent transpose convolutional layers are used in the decoder for upsampling $DUT = N$ instead of time distributed 2D transpose convolutions.

5 Conclusion

We have successfully explored and analysed various Convolutional Recurrent models for the task of video anomaly detection. We compare the performance of the proposed models with changes in their configurations to help pick the most suitable candidate models and to make concrete design choices for the task at hand. Moreover, we have shown that the performance of ConvGRU models are mostly better than ConvLSTM at a lower computational cost, making it the most feasible option in contrast to what we have seen in the literature. For this research, we provide detailed quantitative and qualitative analysis on the performance of the models on several benchmark video anomaly detection datasets with detailed analysis and discussion on results that confirm our hypotheses. Since our current work focused on two main hyper-parameters such as the number of recurrent layers and types of upsampling layers, as future work we intend to study the effects of other trivial hyper-parameters such as input resolution, time steps, number of layers along with exploring hybrid models that can reconstruct and predict frames based on the learnt compact representation. The effect of gray-scale versus color input channels of video frames on the performance of ConvGRU variants is to be evaluated as well in the future. We also intend to analyse the effectiveness of the proposed methods in other video-related tasks such as action recognition, captioning and event classification. We believe that our study will assist developers and researchers in choosing the adequate architecture for applications involving learning representations of and from videos.

Authors' contributions AR was responsible for designing the experiments, conducting and analysing them and writing of the manuscript. FK was responsible for overview, guidance, review and suggestions over the experiments and contents of the manuscript. Both authors read and approved the final manuscript.

Funding The authors declare that they received NO financial support for the research, authorship, and/or publication of this article.

Data availability The authors declare that the datasets analysed during the current study are publicly available from [38–40]. The instructions for download and the terms of use of these datasets are available in the corresponding original papers.

Code availability The authors declare that the code for the experiments conducted in this study is available at the following GitHub repo https://github.com/ambareeshravi/Thesis_VideoAnomalyDetection/.

Declarations

Competing interests The authors declare that they have NO affiliations with or involvement in any organization or entity with any financial or non-financial interest in the subject matter or materials discussed in this manuscript.

References

1. Garcia-Garcia A, Orts-Escolano S, Oprea S, Villena-Martinez V, Martinez-Gonzalez P, Garcia-Rodriguez J. A survey on deep learning techniques for image and video semantic segmentation. Appl Soft Comput. 2018;70:41–65.
2. Nadeem MS, Franqueira VNL, Zhai X, Kurugollu F. A survey of deep learning solutions for multimedia visual content analysis. IEEE Access. 2019;7:84003–19.
3. Suarez JJP, Naval Jr PC. A survey on deep learning techniques for video anomaly detection. arXiv preprint. arXiv:2009.14146; 2020.
4. Collins RT, Lipton AJ, Kanade T. Introduction to the special section on video surveillance. IEEE Trans Pattern Anal Mach Intell. 2000;22(8):745–6.
5. Sultani W, Chen C, Shah M. Real-world anomaly detection in surveillance videos. In: 2018 IEEE/CVF conference on computer vision and pattern recognition. 2018. p. 6479–88.
6. Nguyen TN, Meunier J. Anomaly detection in video sequence with appearance-motion correspondence. In: Proceedings of the IEEE/CVF international conference on computer vision. 2019. p. 1273–83.
7. Hao W, Zhang R, Li S, Li J, Li F, Zhao S, Zhang W. Anomaly event detection in security surveillance using two-stream based model. Secur Commun Netw. 2020. https://doi.org/10.1155/2020/8876056.
8. Liu K, Zhu M, Fu H, Ma H, Chua TS. Enhancing anomaly detection in surveillance videos with transfer learning from action recognition. In: Proceedings of the 28th ACM international conference on multimedia, MM '20. New York: Association for Computing Machinery; 2020. p. 4664–8.

9. Pittino F, Puggl M, Moldaschl T, Hirschl C. Automatic anomaly detection on in-production manufacturing machines using statistical learning methods. Sensors. 2020;20(8):2344.
10. Fernando T, Gammulle H, Denman S, Sridharan S, Fookes C. Deep learning for medical anomaly detection—a survey. Preprint. arXiv:2012.02364; 2020.
11. Chalapathy R, Chawla S. Deep learning for anomaly detection: a survey. Preprint. arXiv:1901.03407; 2019.
12. Kiran BR, Thomas DM, Parakkal R. An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. J Imaging. 2018;4(2):36.
13. Bengio Y, Lamblin P, Popovici D, Larochelle H, et al. Greedy layer-wise training of deep networks. Adv Neural Inf Process Syst. 2007;19:153.
14. Ribeiro M, Lazzaretti A, Lopes H. A study of deep convolutional auto-encoders for anomaly detection in videos. Pattern Recognit Lett. 2018;105:13–22.
15. An J, Cho S. Variational autoencoder based anomaly detection using reconstruction probability. Special Lect IE. 2015;2(1):1–18.
16. Chen Z, Yeo CK, Lee BS, Lau CT. Autoencoder-based network anomaly detection. In: 2018 wireless telecommunications symposium (WTS). IEEE; 2018. p. 1–5.
17. Zhao Y, Deng B, Shen C, Liu Y, Lu H, Hua XS. Spatio-temporal autoencoder for video anomaly detection. In: Proceedings of the 25th ACM international conference on multimedia. 2017. p. 1933–41.
18. Baur C, Wiestler B, Albarqouni S, Navab N. Deep autoencoding models for unsupervised anomaly segmentation in brain MR images. In: International MICCAI brainlesion workshop. Springer; 2018. p. 161–9.
19. Hasan M, Choi J, Neumann J, Roy-Chowdhury AK, Davis LS. Learning temporal regularity in video sequences. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. p. 733–42.
20. Nguyen TN, Meunier J. Hybrid deep network for anomaly detection. Preprint. arXiv:1908.06347; 2019.
21. Li Z, Li Y, Gao Z. Spatiotemporal representation learning for video anomaly detection. IEEE Access. 2020;8:25531–42.
22. Nayak R, Pati UC, Das SK. A comprehensive review on deep learning-based methods for video anomaly detection. Image Vis Comput. 2020. https://doi.org/10.1016/j.imavis.2020.104078.
23. Zhu S, Chen C, Waqas S. Video anomaly detection for smart surveillance. Preprint. arXiv:2004.00222. 2020.
24. Doshi K, Yilmaz Y. Continual learning for anomaly detection in surveillance videos. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. 2020. p. 254–5.
25. Sultani W, Chen C, Shah M. Real-world anomaly detection in surveillance videos. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018. p. 6479–88.
26. Tran D, Bourdev L, Fergus R, Torresani L, Paluri M. Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE international conference on computer vision. 2015. p. 4489–97.
27. Ji S, Wei X, Yang M, Kai Y. 3d convolutional neural networks for human action recognition. IEEE Trans Pattern Anal Mach Intell. 2012;35(1):221–31.
28. Donahue J, Anne Hendricks L, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T. Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2015. p. 2625–34.
29. Shi X, Chen Z, Wang H, Yeung DY, Wong WK, Woo WC. Convolutional LSTM network: a machine learning approach for precipitation nowcasting. Preprint. arXiv:1506.04214; 2015.
30. Ranzato M, Szlam A, Bruna J, Mathieu M, Collobert R, Chopra S. Video (language) modeling: a baseline for generative models of natural videos. Preprint. arXiv:1412.6604; 2014.
31. Srivastava N, Mansimov E, Salakhudinov R. Unsupervised learning of video representations using LSTMS. In: International conference on machine learning. PMLR; 2015. p. 843–52.
32. Luo W, Liu W, Gao S. Remembering history with convolutional LSTM for anomaly detection. In: 2017 IEEE international conference on multimedia and expo (ICME). IEEE; 2017. p. 439–44.
33. Medel JR, Savakis A. Anomaly detection in video using predictive convolutional long short-term memory networks. Preprint. arXiv:1612.00390; 2016.
34. Chong YS, Tay YH. Abnormal event detection in videos using spatiotemporal autoencoder. In: International symposium on neural networks. Cham: Springer; 2017. p. 189–96.
35. Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature. 1986;323(6088):533–6.
36. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–80.
37. Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. Preprint. arXiv:1406.1078; 2014.
38. Lu C, Shi J, Jia J. Abnormal event detection at 150 fps in matlab. In: Proceedings of the IEEE international conference on computer vision. 2013. p. 2720–7.
39. Mahadevan V, Li W, Bhalodia V, Vasconcelos N. Anomaly detection in crowded scenes. In: 2010 IEEE computer society conference on computer vision and pattern recognition. IEEE; 2010. p. 1975–81.
40. Adam A, Rivlin E, Shimshoni I, Reinitz D. Robust real-time unusual event detection using multiple fixed-location monitors. IEEE Trans Pattern Anal Mach Intell. 2008;30(3):555–60.
41. Kozlov Y, Weinkauf T. Persistence1d: extracting and filtering minima and maxima of 1d functions. http://people.mpi-inf.mpg.de/weinkauf/notes/persistence1d.html. 2015. p. 11–01.

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