ANOMALY DETECTION IN SURVEILLANCE VIDEO

Karan Thakkar[1]
Student of Information Technology
Mumbai University
Mumbai, India

Kuldeep Kadiya[2]
Student of Information Technology
Mumbai University
Mumbai, India

Mitiksh Suthar[3]
Student of Information Technology
Mumbai University
Mumbai, India

Mr. Jigar Chauhan[4]
Assistant Professor of Information Technology
Mumbai University
Mumbai, India

***Abstract*—**Urban areas are presently picking to arrangement camera-based reconnaissance framework for observing distinctive territory. This reconnaissance model is exceptionally subject to human cooperation which can prompt human blunder. Our observation framework diminishes the degree of human communication by killing the need to continually screen the video feed. We use LSTM auto encoder to recognize irregularity in video information. We train a LSTM auto encoder on just Normal dataset, so when an input information that have various features from Normal dataset are taken care of to the model the comparing recreation anomaly will increment. We call such input information "abnormal data".

Keywords—Abnormal Events, Neural Network, Reconstruction, Spartial Network

I. INTRODUCTION

Envision we have a huge number of reconnaissance cameras that work constantly, a portion of these cameras are mounted in distant zones or roads where it's far-fetched that something unsafe would happen, others are introduced in packed roads or city squares. There is a wide assortment of unusual occasions that may occur even in a solitary area, and the meaning of abnormal occasion varies from area to another and now and again.

Significant occasions that are of interest in long video successions, like surveillance film, regularly have an amazingly low Probability of happening. Utilizing automated system to identify unusual occasions in this situation is profoundly alluring and prompts better security and more extensive observation. In general, the way toward identifying irregular occasions in recordings is a difficult issue that at present draws in much consideration by specialists, it additionally has wide applications across industry verticals, and as of late it has gotten one of the fundamental errands of video analysis. There is a huge interest for building up an anomaly identification approach that is quick and exact in real-world applications.

Video data make modelling and representation troublesome, it can be because of high dimensionality, Noise in video and high variety of occasions. Anomalies are exceptionally subject to context, for Example driving on street is ordinary however driving on trial is unusual. Irregularity is subject to eyewitness; some may think a specific activity is dubious and some may not. This is the principal limit for anomaly detection utilizing ML.

However, these strategies simply relevant to labelled video footages where events of revenue are obviously characterized and doesn't include profoundly impeded scenes, like swarmed scenes. Moreover, the cost of labelling each sort of event is amazingly high. All things considered, it isn't ensured to cover each past and future events. The recorded video film is likely not long enough to catch a wide range of activities, particularly abnormal activities which rarely or never happened.

There are different successful cases in this field of activity recognition. In However, these techniques are simply applicable to labelled data and the event of interest is unmistakably characterized. Likewise, these strategies can't work as expected if the scenes in input data changes, for instance, on the off chance that these models are prepared to detect collision, they can't recognize collision when there is abrupt change is climate.

This paper presents a system design for utilizing deep neural network for anomaly detection in reconnaissance video, this video is gathered consequently from a long footage through deep learning approach. Deep neural network made out of a stake of auto encoder to handle video frames in unsupervised way that catches spatial structure in video data. At that point reconstruction blunder is determined from the test data to decide if the data have anomaly or it's a normal footage. Our proposed strategy isn't domain specific (i.e., It's not specific to particular task), diminish human exertion, and can easily applied to various scene.

II. RELATED WORK

In January 2018, L. Wang, F. Zhou, Z. Li, W. Zuo and H. Tan made an anomalies detection on video Convolutional layers have shown promise in recent applications of convolutional neural networks for object detection and recognition, especially in photos. Convolutional neural networks are supervised and involve learning signals in the form of labels. A spatiotemporal architecture for detecting anomalies in images, including crowded scenes, is proposed. The architecture was made of using two main components, one is for identifying Our architecture is made up of two main components: one for identifying spatial features and the other for learning the temporal evolution of those features. Experiments on the Avenue, Subway, and UCSD benchmarks prove that our method's detection accuracy is comparable to state-of-the-art methods at up to 140 frames per second. [1]

In 2017, Y. Jia, C. Zhou and M. Motani they suggest an unsupervised deep learning scheme called a spatio-temporal autoencoder (STAE) for learning features from large-scale and high-dimensional patient data with missing observations. STAE can automatically recognize patterns and dependencies in patient data, even with missing values, and learn a compact representation of each patient for better classification, according to both spatial and temporal encoding. To test and verify the performance of STAE, we extract an EEG data set from the publicly available UCI ML Repository. [2]

III. METHODOLOGY

The techniques depicted here work on the way that when any video sequence of unusual occasion is passed to auto encoder prepared on normal video won't able to build the abnormal video. On the off chance that the subsequent reconstruction blunder is higher than input data contains unusual data or the other way around. The auto encoder is trained on normal data which is produced from CCTV film. The end-to-end model comprises of temporal encoder and decoder which take in examples and learn pattern extract feature from volume video data. The model is prepared with video volumes comprises of just normal scenes, with the goal to limit the reconstruction error between the input video volume and the output video volume reconstructed by the learned model. After the model is appropriately prepared, normal video volume is relied upon to have low reconstructed error, though video volume comprising of unusual scenes is required to have high reconstructed error. By thresholding on the error delivered by each testing input volumes, our system will actually want to recognize when an unusual thing happens [1].

3.1 Pre-processing

The primary task of pre-processing is to change raw information over to adequate input for the model. Separation the training video frames into temporal sequences, every one of size 10 utilizing the sliding window strategy. Video is partitioned in to frames and frame resized to 236 x 236. Each pixel esteem is scaled somewhere in the range of 0 and 1 by isolating every pixel with 236. As the quantity of parameters is tremendous with required more data for training model, consequently we use data augmentation in the temporal dimension. To produce additional training sequences, we connect frames with different skipping steps. For e.g., the primary step 1 sequence is comprised of frames (1, 2, 3, 4, 5, 6, 7, 8, 9, 10), while the principal step 2 grouping comprises of frames (1, 3, 5, 7, 9, 11, 13, 15, 17, 19) [1].

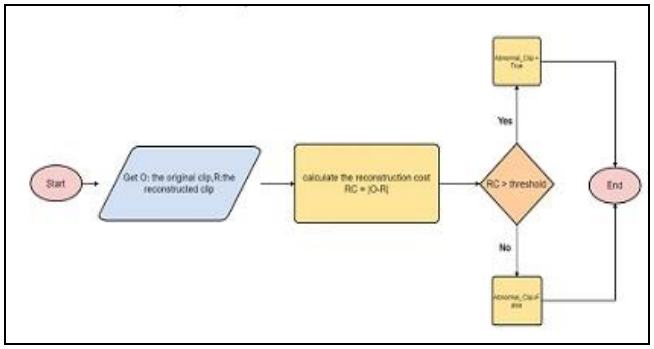

Figure 1. System Diagram

3.1.1 Auto encoder

Auto encoders, as the name proposes, comprise of two phases: encoding and decoding. It was first used to diminish dimensionality by setting the quantity of encoder output units not exactly as the input. The model is typically prepared utilizing back-propagation in an unsupervised way, by limiting the reconstruction error of the decoding results from the input sources. With the activation function picked to be nonlinear, an auto encoder can extricate more valuable features than some basic linear transformed techniques like PCA [1].

The auto encoder comprises of two phases, encoder and decoder. Encoder is Capable of learning productive portrayals of the input information (x) called the encoding f(x). The last layer of the encoder is known as the bottleneck, which contains the input representation f(x). Decoder creates a reconstruction of the input info r = g (f(x)) utilizing the encoding in the bottleneck. The encoder comprises of 2 distinctive encoder, spatial encoder and temporal encoder

Three instances of utilizations of Auto Encoders are given below:

- Data Storage: The encoding measures can down huge amounts of information, compacting it. This cycle, as you can imagine, has huge advantages for data stockpiling at scale.
- Feature identification: The process used to encode the data recognizes features of the data that can be utilized to distinguish it. This rundown of features is utilized in numerous systems to comprehend the data. (Convolutional Neural Networks likewise feature identification in pictures)
- Recommendation systems: One utilization of auto encoders is in this is the systems that recognize movies or TV series you are probably you like on stream service.

Input is taken care of in first layer of spatial encoder, this convolution layer comprises of 128 filters with kernel size of 11x11 and step 4. The input is bunch of 10 greyscale picture with goal resolution 236x236. The output of this layer is encoded features.

3.1.2 Spatial encoder

The spatial auto encoder in STAE is an exemplary encoder decoder design [3], [4] dependent on multi-layer neural n/w. The encoder E comprises of a multi-facet neural n/w, where each hidden layer of the n/w is prepared to deliver a more elevated level representation of the input information. This is finished by upgrading a local unsupervised model dependent on the information it gets from the past layer. In this way, every layer creates a representation of the input design that is more unique than the past layer. The decoder D mirrors the formation of the encoder [2].

3.1.3 Temporal (Long Short-term Memory) Auto encoder

After spatial encoding, the encoded information Y are pass into a temporal auto encoder to extricate the temporal features. In STAE, the temporal auto encoder is built by a bunch of LSTM cells, which is a unique sort of RNN. RNN is a sequence dependent neural n/w as it considers the current input just as the choice made in the previous time step. It is powerful in learning in features from time series information with temporal data. LSTM is a particular RNN which has long-momentary memory units in the hidden layer. It is explicitly intended for long sequences. LSTM units have a capacity to alternatively keep specific data from the past state, and to pick certain data on the present status to be updated [2].

3.1.4 Convolutional Neural Networks (CNNs / ConvNet)

Convolutional Neural Networks are basically the same as ordinary Neural Networks from the past section: they are comprised of neurons that have learnable weights and biases. Every neuron gets a few information inputs, plays out a dot product and alternatively follows it with a non-linearity. The entire network actually expresses a solitary differentiable score work: from the raw picture pixels on one side to class scores at the other. They actually have a loss function (for example SVM/Softmax) on the last (completely-connected) layer and every one of the tips/stunts we created for learning regularly Neural Networks actually apply. ConvNet models make the unequivocal assumption that the input information are pictures, which permits us to encode certain properties into the architecture. These make forward function more efficient to reduce the parameter in network [5].

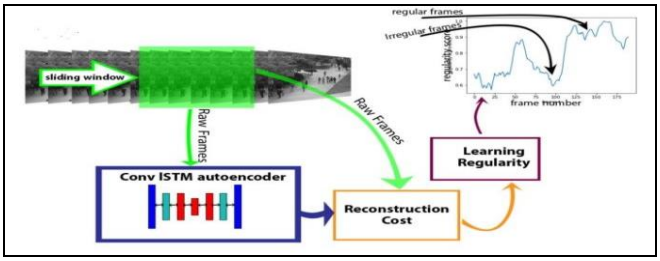

Figure 2. Training Model Architecture

3.1.5 Long Short-term Memory (LSTM)

LSTM is variant of RNN. Long Short Term Memory (LSTM) model which incorporates a recurrent gate called forget gate. With the new structure, LSTMs prevent backpropagated errors from vanishing or exploding, thus can work on long sequences and they can be stacked together to capture higher level information. The formulation of a typical LSTM unit is summarized with Figure 3 and equations (1) through (6) [2].

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

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

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

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

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

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

$$h_t = o_t \otimes tanh(C_t) \quad (6)$$

Figure 3. Equations

3.1.6 Convolutional LSTM

A variant of the LSTM architecture, namely Convolutional Long Short-term Memory (ConvLSTM) model was introduced by Shi et al. in [7] and has been recently utilized by Patraucean et al. in [6] for video frame prediction. Compared to the usual fully connected LSTM (FC-LSTM), ConvLSTM has its matrix operations replaced with convolutions. By using convolution for both input-to-hidden and hidden-to-hidden connections, ConvLSTM requires fewer weights and yield better spatial feature maps. The formulation of the ConvLSTM unit can be summarized with (7) through (12)

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

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

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

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

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

$$h_t = o_t \otimes tanh(C_t) \quad (12)$$

Figure 4. Equations

While the equations are similar in nature to (1) through (6), the input is fed in as images, while the set of weights for every connection is replaced by convolutional filters (the symbol ↞ denotes a convolution operation). This allows ConvLSTM work better with images than the FC-LSTM due to its ability to propagate spatial characteristics temporally through each ConvLSTM state [2].

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

IV. ARCHITECTURE

Regular Neural Nets. As we found in the previous section, Neural Networks get an input information (a single vector), and change it through a progression of hidden layers. Each hidden layer is comprised of a bunch of neurons, where every neuron is completely associated with all neurons in the past layer, and where neurons in a single layer work totally autonomously and don't share any associations. The last completely-connected layer is known as the "output layer" and in classification settings it addresses the class scores.

Regular Neural Nets don't scale well to full pictures. In CIFAR-10, images are just of size 32 x 32 x 3 (32 wide, 32 high, 3 shading channels), so a single completely-connected neuron in a first layer of a Neural Network would have 32*32*3 = 3072 loads or weights. This sum actually appears to be reasonable, clearly this completely connected structure doesn't scale to bigger pictures. For instance, a picture of more decent size, for example 200x200x3, would prompt neurons that have 200*200*3 = 120,000 loads. Additionally, we would in all likelihood need to have a few such neurons, so the boundaries would add up rapidly! Plainly, this full network is inefficient and the tremendous number of boundaries would rapidly prompt overfitting [5].

3 Dimension volumes of neurons. Convolutional Neural Networks exploit the way that the information comprises of pictures and they oblige the design in a more reasonable manner. Specifically, irregular Neural Network, the layers of a ConvNet have neurons organized in 3 measurements: width, height, depth. (Note that the word depth here alludes to the third dimension of an activation volume, not to the depth of a full Neural Network, which can allude to the all layers in an n/w.) For e.g., the input pictures in CIFAR-10 are an n/p volume of activations, and the volume has measurements 32x32x3 (width, height, depth individually). As we will before see, the neurons in a layer may be associated with a little region of the layer, rather than the entirety of the neurons in a completely-connected way. Besides, the last output layer would for CIFAR-10 have measurements 1x1x10. Finish of the ConvNet architecture we will decrease the full image into a single vector of class scores, orchestrated along the depth dimension. Here is a perception:

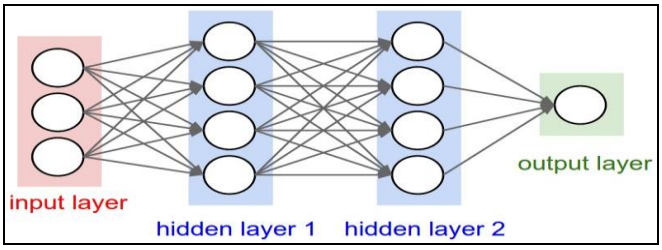

Figure 5. A regular 3-layer Neural Network.

A ConvNet organizes its neurons in three proportions (width, height, depth), as seen in one of the layers. Each layer of a ConvNet changes the 3 Dimension input volume to a 3 Dimension output volume of neuron activations. In this model, the red input layer holds the image, width and height is image dimensions, and the depth is 3 (Red, Green, Blue channels). A ConvNet is comprised of Layers. Each Layer has a basic API: It changes an input 3D volume to an output 3D volume with some differentiable function that might have parameters [5].

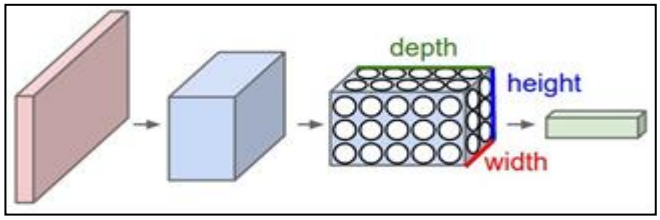

Figure 6. Neurons in three Dimensions

Layers used to construct ConvNet

1. Convolution layer

2. Pooling layer

3. Fully-connected layer

Detail a simple ConvNet for CIFAR-10 classification could have the architecture [INPUT - CONV - RELU - POOL - FC]

- INPUT [32x32x3] hold the pixel values of the image, here height and width 32, 32 respectively, and three channel color R, G, and B.
- CONV layer process the neurons from output that are connected to local region of input, computing dot product between small region connected to input volume and their weights. As result volume such as [32x32x12] we decide to use 12 filters.

- RELU layer used to apply activation function to an elementwise such as the max (0, x) max (0, x) at zero threshold. This leaves unchanged size of volume ([32x32x12]).
- POOL layer will perform a down sampling operation along the spatial dimensions (width, height), resulting in volume such as [16x16x12].
- FC (i.e., fully-connected) layer will estimate the class scores, result about volume of size [1x1x10], where every one of the 10 numbers relate to a class score, such as among the 10 classifications of CIFAR-10. as with ordinary Neural Networks and as the name suggests, every neuron in this layer will be associated with every one of the numbers in the previous volume

This way, ConvNet change the original image layer by layer from the original pixel esteems to the last class scores. Note that a few layers contain parameters and other don't. Specifically, the CONV/FC layers perform transformation. That are function of activation in i/p volume and parameters (biases of neurons & weights). RELU/POOL layers will implement a fixed function. The parameters in the CONV/FC layers will be prepared with gradient descent so the class scores that the ConvNet uniform with the labels in the training set for each picture [5].

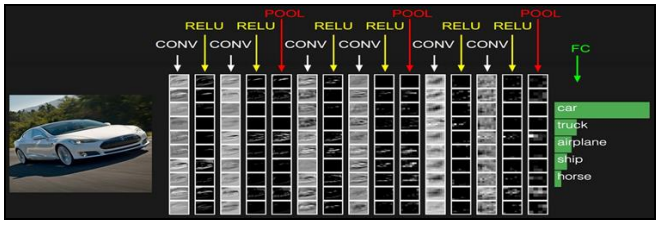

Figure 7. ConvNet CIFAR-10

V. DATA SET

Our datasets, and Subway datasets are the five most widely used benchmarking datasets on which we train our model. For each dataset, all videos are taken from a fixed location. There are no unusual incidents in any of the training recordings. Both usual and irregular occurrences can be seen in testing recordings [1].

There are 16 preparation and 21 research video clips in the dataset. The length of each clip varies from under a minute to over two minutes. People walking between the stairwell and the subway entrance constitute usual scenes, while irregular activities include people running, walking in the opposite direction, loitering, and so on. Camera shakes and a few outliers in the training data are among the dataset's difficulties. In addition, a typical pattern occurs infrequently in the training results. Each of the 34 training and 36 testing video clips in the one of the dataset contains 200 frames [1].

The videos show people walking towards and away from the camera in groups. The second part of dataset contains 16 training and 12 testing video clips, each with a different number of frames. The videos show people walking parallel to the camera plane. Bikers, skaters, wheelchairs, and people walking in the grass are among the anomalies in the two datasets. The subway entrance dataset is 1 hour 30 minutes long and contains 65 unusual events divided into five categories: walking in the wrong direction, no payment, loitering, odd interpersonal encounters, and miscellaneous (e.g., suddenlly stop, run fast).

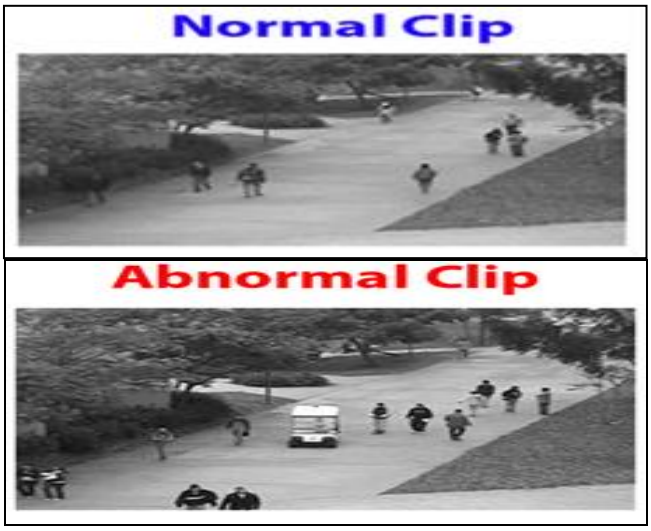

Figure 8. Dataset Frames

The first minutes of the video was dedicated to instruction. The exit dataset is 40 minutes long and contains 18 odd events divided into three categories: walking in the wrong direction, loitering , and miscellaneous (e.g., abrupt halt, looking around, janitor cleaning the wall, getting off the train and quickly getting back on the train). The first couple of minutes of the video are dedicated to preparation.

First, let's take a look at first datasets test 32. There is a bicycle on the walkway at the start of the video, which explains the low regularity score. The regularity score begins to rise after the bicycle has left. Second bicycle joins at frame number 59-60, and the score reduce before increasing immediately after it leaves.

A skater enters the walkway at the start of the video, and someone steps on the grass at frame 135-140, which explains the two decreases in the regularity score dataset. A small cart crosses the walkway causing a decrease in the regularity value. After the cart left, the regularity score returns to normal. Two bicycles cross the walkway in of the dataset, one at the beginning and the other at the end of the video.

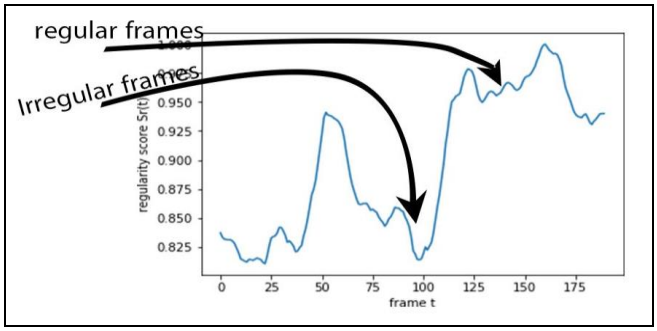

Figure 9. Result Graph

VI. CONCLUSION

Use a variety of datasets or collect your own data or Make your own data with a surveillance camera or camera in your space. As the training data videos that only contain daily events, it is relatively simple to obtain. Combine different datasets and see if the model still works. to speed up the process of identifying anomalies use few sequences in the stages.

VII. REFERENCES

[1] L. Wang, F. Zhou, Z. Li, W. Zuo and H. Tan, "Abnormal Event Detection in Videos Using Hybrid Spatio-Temporal Autoencoder," 2018 25th IEEE International Conference on Image Processing (ICIP), 2018, pp. 2276-2280, doi: 10.1109/ICIP.2018.8451070.

[2] Y. Jia, C. Zhou and M. Motani, "Spatio-temporal autoencoder for feature learning in patient data with missing observations," 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 2017, pp. 886-890, doi: 10.1109/BIBM.2017.8217773.

[3] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in Proc. ICML, 2008, pp. 1096–1103.

[4] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," J. Mach. Learn. Res., vol. 11, pp. 3371–3408, Dec. 2010.

[5] https://cs231n.github.io/convolutional-networks/

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

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