**Anomaly Detection in Videos using SpatioTemporal Autoencoders**

Priyesh Kaushik | [kaushik.pr@husky.neu.edu](mailto:kaushik.pr@husky.neu.edu)

Haroon Perveez | parveez.s@husky.neu.edu

Neelam Babel | babel.n@husky.neu.edu

**Objective**

Recent applications of convolutional neural networks have shown promises of convolutional layers for object detection. However, convolutional neural networks are supervised and require labels as learning signals. As part of this project, we propose a spatiotemporal architecture for anomaly detection in videos including crowded scenes. Our architecture includes two main components, one for spatial feature representation, and one for learning the temporal evolution of the spatial features.

**Pipeline Design**

Our approach consists of 2 main stages :

**Step 1 - Preprocessing** : Each frame is extracted from the raw videos and resized into 227\*227. The images are then converted to grayscale to reduce dimensionality. The processed images are then normalized to have a zero mean and unit variance. The input to the model is video volumes, where each volume contains 10 consecutive frames with various skipping strides. To generate these volumes, we concatenate frames with stride-1, stride-2 and stride-3. The first stride-1 sequence is made up of frame {1,2,3,4,5,6,7,8,9,10} whereas the first stride-2 sequence contains frame number {1,3,5,7,9,11,13,15,17,19} and the stride-3 sequence would contain the frame number {1,4,7,10,13,16, 19,22,25,28}. This input goes to the training model.

**Step 2 – Building and Training the model** : Our model consists of two parts – spatial autoencoder (fig 1.1) for learning spatial features of each video frame, and a temporal encoder (fig 1.2) for learning the temporal patterns of the encoded spatial features.

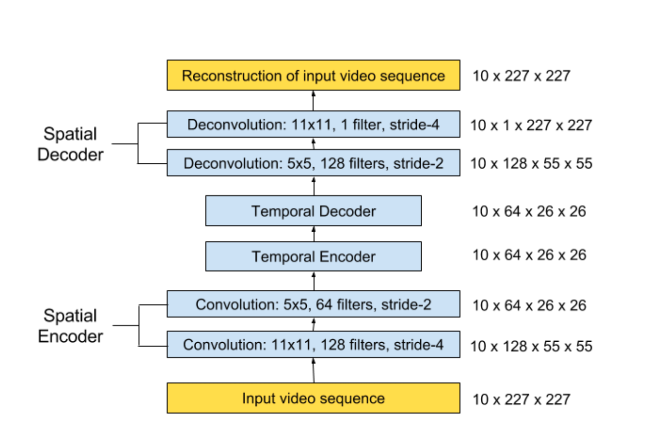


fig 1.1

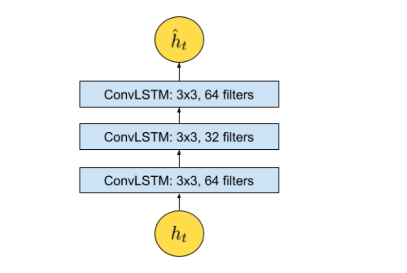


fig 1.2

We then pass the training data (16 videos) to our model.

**Implementation Details**

**Data** - We are using the Avenue Dataset for training and testing our model. The dataset can be downloaded using the link : <http://www.cse.cuhk.edu.hk/leojia/projects/detectabnormal/dataset.html>

**Training Model** – We are using Spatial Temporal Convolution Technique to detect anomalies. In our model, spatial encoder and decoder have two convolutional and deconvolutional layers respectively, and the temporal encoder is a 3 layer convolutional LSTM model. The model is trained using back-propagation in an unsupervised manner, by minimizing the reconstruction error of the decoded results from the original inputs.

Convolution preserves the spatial relationship between pixels by learning image features using small squares of input data. Furthermore, a ConvLSTM has it’s matrix operations replaced with convolutions. By using convolution for both input-to-hidden and hidden-to-hidden cnnections, ConvLSTM requires a fewer weights and can propagate spatial characteristics temporally through each ConvLSTM state and yield better spatial feature maps.

Model Parameters : We are using Adam optimizer and it will be setting the learning rate automatically based on the models weight update history. We are using mini batches of size 64, and each training volume is trained for a maximum of 50 epochs. Also, we are using hyperbolic tangent as the activation function of the spatial encoder and decoder.

**Reconstruction Error** – The reconstruction error of all the pixel values in a frame of the video sequence is taken as the Euclidean distance between the input frame and the reconstructed frame.

**Anomaly Detection –** We use the reconstruction error of each frame to determine whether the frame is normal or anomalous. The threshold value determines the sensitivity of the model. Example, setting a low threshold will make our model more sensitive to the happenings in the scene, so that more alarms could be triggered even on the smallest of changes.

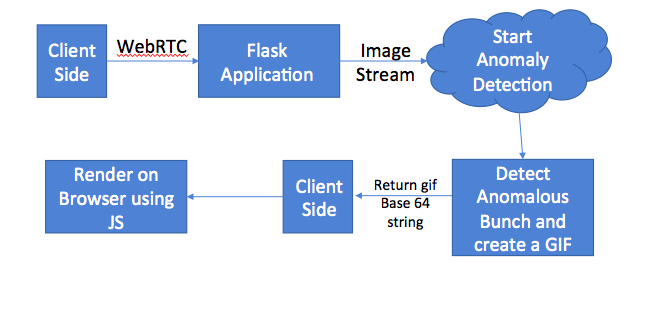
**Analysis of Our Model**

Anomalous events and false alarm counts as detected by our model :

****

**How to deploy and run the model**

* The video uploaded is converted into frames using JavaScript and the blob files of 10 frames at a time are sent to the server with a POST request
* The images are then converted in numpy Arrays which are feeded into the model
* Anomalous Bunch is detected by calculating the reconstruction error and thresholding
* A gif is created from the anomalous frames and encoded to Base64 string which is rendered on the browser using JavaScript



**References**

https://arxiv.org/ftp/arxiv/papers/1612/1612.00390.pdf