CRIMINAL ACTIVITY DETECTION USING DEEP LEARNING APPROACH

Kapil Sharma [*1], Prabhat Singh[*2], Lakshya Kathayat[*3] , Chaitanya Prakash Singh Jayara[*4], Anjani Gupta[*5]

[*1]Student,B.Tech,I.T.,Dr.Akhilesh Das Gupta Institute of Technology & Management, Delhi, India

[*2]Student,B.Tech,I.T.,Dr.Akhilesh Das Gupta Institute of Technology & Management, Delhi, India

[*3]Student,B.Tech,I.T.,Dr.Akhilesh Das Gupta Institute of Technology & Management, Delhi, India

[*4]Student,B.tech,I.T.,Dr.Akhilesh Das Gupta Institute of Technology & Management, Delhi, India

[*5]Asst. Professor,I.T.,Dr.Akhilesh Das Gupta Institute of Technology & Management, Delhi, India

ABSTRACT

Circuit Television Cameras (CCTV's) are widely used to control occurrence of crimes in the surroundings. Although CCTVs are deployed at various public and private areas to monitor the surroundings there is no improvement in the control of crimes. This is because CCTV requires human supervision which may lead to human prone errors like missing of some important crime events by human while monitoring so many screens recorded by CCTVs at same time. To overcome this issue, we came up with Crime Intension Detection System that detects crime in real time videos, images and alerts the human supervisor to take the necessary actions. To alert the supervisors or nearby police station about the occurrence of crime. This can be achieved by the Anomaly detection using Deep Learning using Autoencoders.

An outlier (Anomaly) is an observation in a data set which appears to be inconsistent with the remainder of that set of data. An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism.

Anomalies are defined as events that deviate from the standard, happen rarely, and don‘t follow the rest of the ―pattern‖.

Examples of anomalies include:

● Large dips and spikes in the stock market due to world events

● Defective items in a factory/on a conveyor belt

● Contaminated samples in a lab

Depending on your exact use case and application, anomalies only typically occur 0.001-1% of the time — that‘s an *incredibly* small fraction of the time**.** The problem is only compounded by the fact that there is a *massive imbalance* in our class labels. By definition, anomalies will *rarely occur,* so the majority of our data points will be of valid events. To detect anomalies, machine learning researchers have created algorithms such as Isolation Forests, One-class SVMs, Elliptic Envelopes, and Local Outlier Factor to help detect such events; however, all of these methods are rooted in *traditional machine learning.*

Keywords: analysis, anamoly, autoencoders, elliptical envelopes, outlier factor.

I. INTRODUCTION

Circuit Television Cameras (CCTV's) are widely used to control occurrence of crimes in the surroundings. Although CCTVs are deployed at various public and private areas to monitor the surroundings there is no improvement in the control of crimes. This is because CCTV requires human supervision which may lead to human prone errors like missing of some important crime events by human while monitoring so many screens recorded by CCTVs at same time. To overcome this issue, we came up with Criminal Activity Detection System that detects crime in real time videos, images and alerts the human supervisor to take the necessary actions. To alert the supervisors or nearby police station about the occurrence of crime. This Crime Detection System can be best implemented by the Anomaly detection using Deep Learning through Autoencoders.

An outlier (Anomaly) is an observation in a data set which appears to be inconsistent with the remainder of that set of data. An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism.

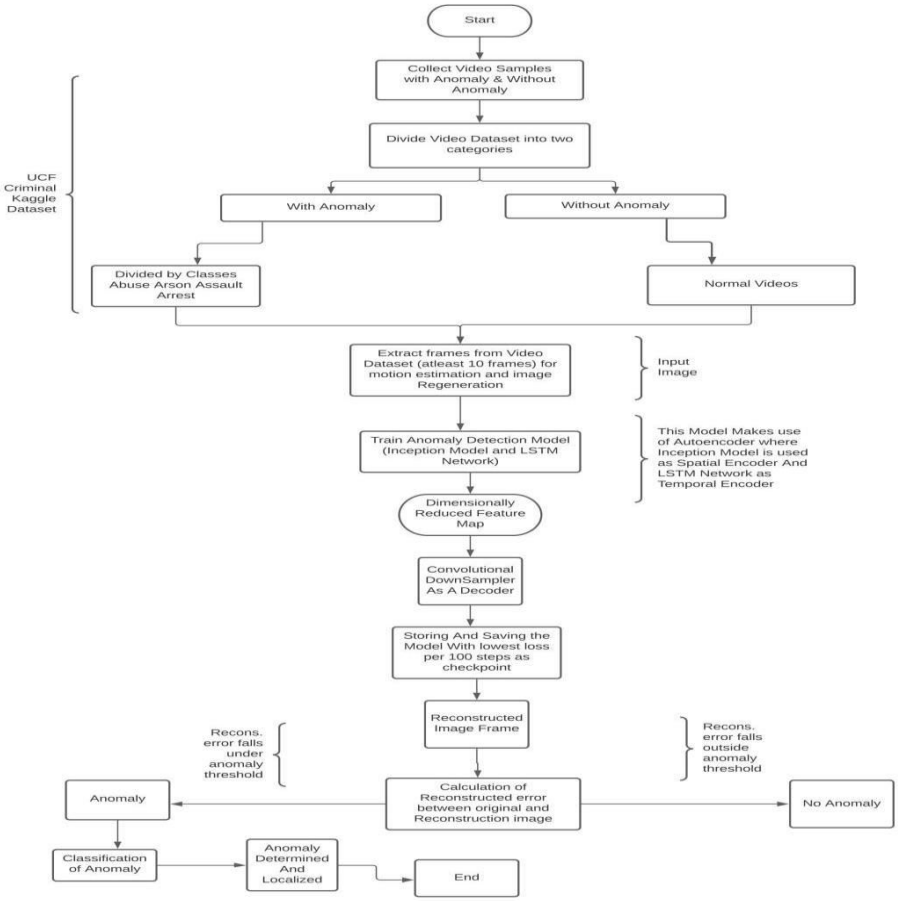
Anomalies are defined as events that deviate from the standard, happen rarely, and don’t follow the rest of the “pattern”.

Examples of anomalies include:

- Unusual pedestrian motion patterns like people walking across a walkway or at the grass surrounding it
- Non-pedestrian entities in the walkway, like bikers, skaters, and small carts.

Depending on your exact use case and application, anomalies only typically occur 0.001- 1% of the time — that's an *incredibly* small fraction of the time**.** The problem is only compounded by the fact that there is a *massive imbalance* in our class labels. By definition, anomalies will *rarely occur,* so the majority of our data points will be of valid events.

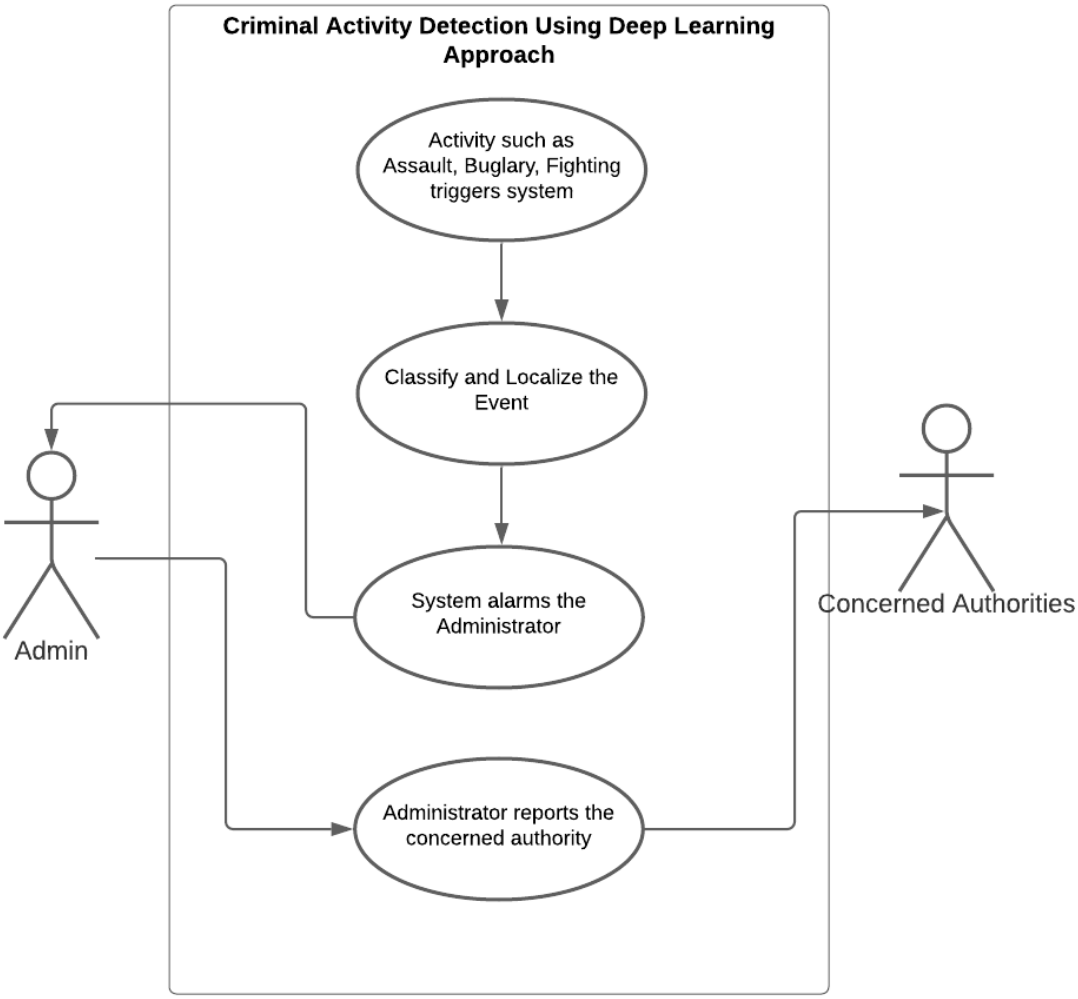
II. METHODOLOGY



First we collected video samples (from UCF Kaggle Criminal Dataset) which are a mixture of videos having Anomaly and not having Anomaly.We then divided dataset manually into 2 categories – With Anomaly and Without Anomaly for training purpose.We categorized the subset one having videos with Anomaly into 4 defined classes as – Abuse,Arson,Assault,Arrest.
Before dataset training, we extracted atleast 10 frames from video each one which were treated as input image for the training process.We then built an autoencoder which was a combination of Inception model and LSTM model used for CNN in motion estimation and image regeneration. Here Inception Model is used as Spatial Encoder i.e., it extracted the specific feature from the image for which it was trained specifically and LSTM model was used as Temporal Encoder i.e., it extracted all the possible features from the frame or image.This did return a dimensionally reduced feature map of the corresponding image.
During dataset training, we used Convolution Decoder to decode the Dimensionally reduced map generated above into a new image which we called a regenerated image. Now iterating over the model, we recorded the checkpoint at each 100 steps as the one with lowest loss in the group of 100. That point was called an epoch. In this way finally a reconstructed image was generated for each frame respectively.

After dataset training,we applied calculation to decide an average value of reconstructed error between each original image and its corresponding reconstructed image.Using the above generated threshold value for reconstructed error we classified the testing data into the respective groups of images having anomaly and images without anomaly - the first one with reconstructed error within the threshold value and the other outside the same range.Now in the first class of images with anomaly mentioned above we further subdivided the images into the above mentioned four classes - Abuse, Arson, Assault and Arrest respectively.Now in each image of the above four classes we located the Anomaly and localize its position for each frame respectively.

III. MODELING AND ANALYSIS



Use Case Diagram

Software Requirements Specification (SRS) is the starting point of the software development activity. Little importance was given to this phase in the early days of software development. The emphasis was first on coding and then shifted to design. Some of the difficulty is due to the scope of this phase. The software project is imitated by the client needs. In the beginning these needs are in the minds of various people in the client organization. The requirement analyst has to identify the requirements by tacking to these people and understanding their needs. In situations where the software is to automated a currently manuals process, most of the needs can be understood by observing the current practice. The SRS is a means of translating the ideas in the minds of the clients (the output) into formal document (the output of the requirements phase). Thus, the output of the phase is a set of formally specified requirements, which hopefully are complete and consistent, while the input has none of these properties.

PERFORMANCE REQUIREMENTS

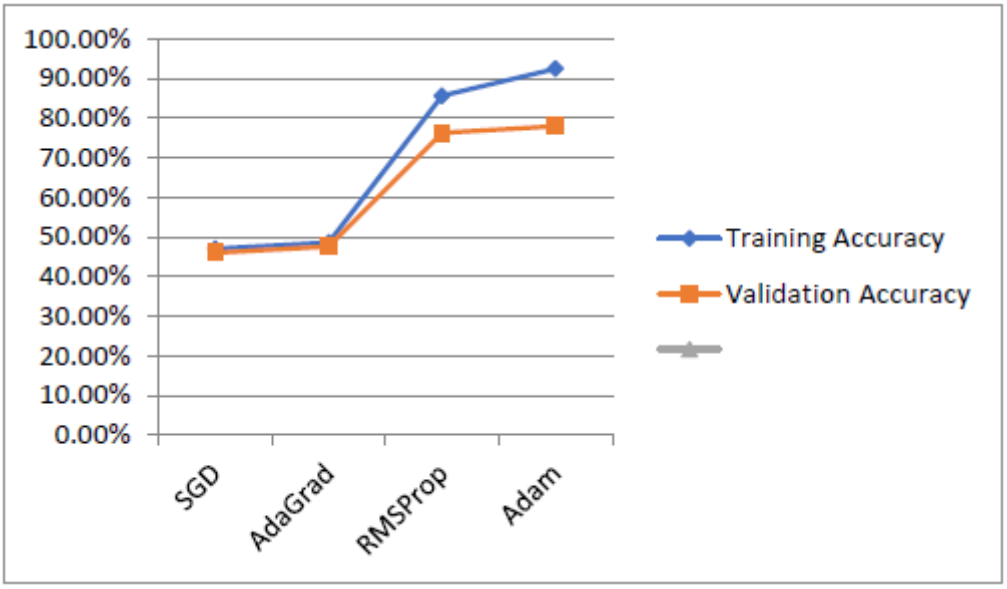
The project must meet the end user requirements. Accuracy and speed must be imposed in the Project. The project is development as easy as possible for the sake of end user. The project has to be developed with view of satisfying the future requirements and future enhancement. The tool has been finally implemented satisfying the needs specified by the company. As per the performance is concerned this system said is performing This processing as well as time taken to generate well reports were also even when large amount of data was used.

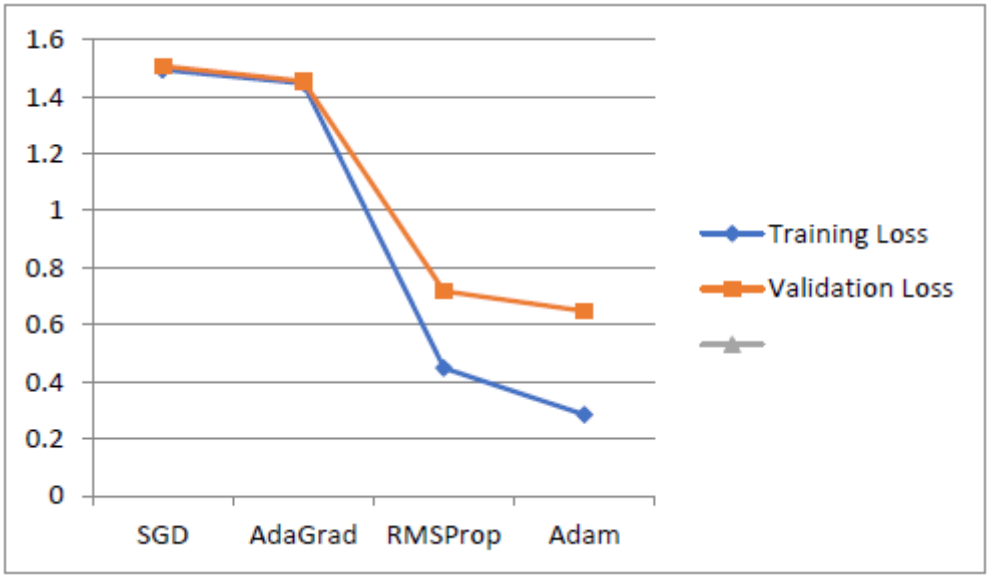
HARDWARE & SOFTWARE REQUIREMENTS

CPU	Intel i5 9400F
GPU	NVIDIA GTX 1050 Ti
IDE/ Code Editor	Jupyter Notebook or Google Collab
Programming Languages	Python3
Frameworks/Models	R-CNN, CNN, Inception v3, LSTM, Auto-Encoders, Open CV, Keras,
Image Processing Tools	GIMP, LabelImg

IV. RESULTS AND DISCUSSION

Optimizer	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss
SGD	47.02%	1.4931	46.03%	1.5064
AdaGrad	48.56%	1.4463	47.58%	1.4537
RMSProp	85.65%	0.4479	76.23%	0.7176
Adam	92.46%	0.2841	77.97%	0.6477

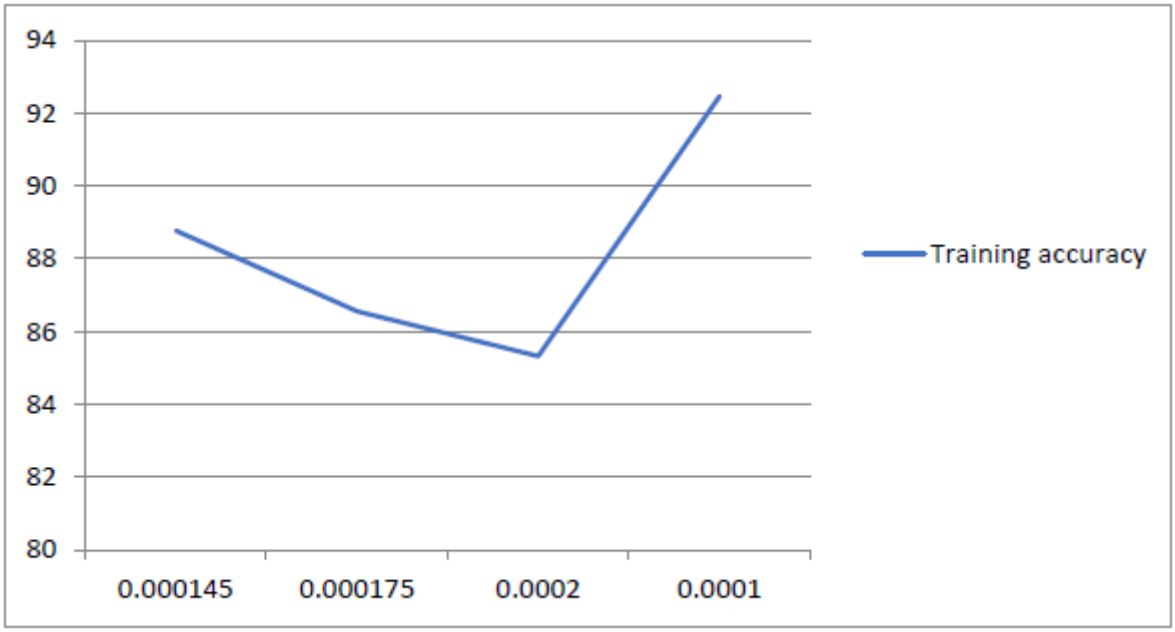




Comparison Graph between Optimizers

Adam Optimizer

Learning rate	Decay rate	Training accuracy	Validation Accuracy
0.000145	0.00002	88.76	72.38
0.000175	0.00001	86.55	70.98
0.0002	0.00003	85.32	69.43
0.0001	**0.00001**	**92.46**	**77.97**



Learning Rate vs Training Accuracy for Adam Optimizer

V. CONCLUSION

All the main points of the research work are written in this section. Ensure that abstract and conclusion should not same. Graph and tables should not use in conclusion.

There are many exciting homes that you can actually get from combining convolutional neural networks (CNN) and recurrent neural networks (RNN). That aggregate uses the exceptional of each worlds, the spatial and temporal worlds. However, RNNs are turing complete, which means they are able to surely study any computational function. This way theoretically we do not want to mix CNNs with RNNs to get matters done, we simply want the RNN. But as usual, in fact we want modules which are specialised for unique signals. The CNNs are proper at handling spatially associated statistics at the same time as the RNNs are proper at temporal signals, it's far simply the no unfastened lunch theorem at play here.Thus the issues wherein a CNN-RNN pair is needed are while dealing with:

- Mapping spatial collection inputs to static output classes.
- Mapping static spatial inputs to squence outputs (speech or textual content captioning)

In the primary case we are able to have a video enter and we want to categorise that video right into a static magnificence category. A specific software might be human-motion recognition, from a series of video frames we want to perceive human motion which includes walking, seated, leaping or running. It might be feasible to have this kind of device perceive on going crimes in actual time and things like detection of riots simply from video feeds. In the second one case, the CNN-RNN pair can feed from, say, a static photograph however output a series output which includes speech or textual content describing the content material of that static photograph. In programs which includes computerized photograph captioning[1] , such structures want to explain the content material of an photograph via way of means of producing photograph captions. In each instances the CNN acts just like the trainable function detector for the spatial signal. It learns effective convolutional functions which operates on a static spatial input(frame) whilst the RNN gets a chain of such high-stage representations to generate an outline of the content material or map to a few static magnificence of outputs. In photograph captioning there's usually an interest mechanism constructed in to permit the device to take care of sure photograph components whilst producing the captions. Facebook is absolutely running on structures which could describe photograph content material to blind humans the usage of such CNN-RNN pairs. Though it's also feasible to do photograph captioning with most effective a totally feedforward neural community structure with smart use of convolutions in time. The use of convolutions in time aren't liable to exploding/vanishing gradient troubles encountered withinside the vanilla

RNNs. Instead of the usage of a normal vanilla RNN one also can alternatively use long-short-time period-memory (LSTM) or gated recurrent unit (GRU) networks for removing the difficulty of long time dependences. Which makes a CNN-LSTM or CNN-GRU pairs extraordinarily effective in managing spatial-temporal signals. We additionally do have the so known as recurrent convolutional neural networks (RCNN) which has recurrent connections without delay withinside the kernels themselves. Though this gadget can be afflicted by incapability to address long-time period dependences simply because the vanilla RNNs.

VI. REFERENCES

[1] https://ieeexplore.ieee.org/abstract/document/7995808

[2] https://openaccess.thecvf.com/content_iccv_2017/html/Lu_Online_Video_ICCV_2017_paper.html

[3] https://ieeexplore.ieee.org/abstrat/document/8624183

[4] https://www.researchgate.net/publication/334279066_A_survey_on_Image_Data_Augmentation_for_Deep_Learning

[5] https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/