ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

ABNORMAL EVENT DETECTION WITH CNN AND UNSUPERVISED CLASSIFICATION

K Madhavi [1], N. Nikhil Varma [2], S. Bindu Madhavi [3], Namrata Singh [4], E. Niveditha [5]
1 Associate Professor, Dept. of CSE, NS Raju Institute of Technology Visakhapatnam, Andhra Pradesh, India
2,3,4,5 Student, Dept of CSE, NS Raju Institute of Technology Visakhapatnam, Andhra Pradesh, India
madhavi.cse@nsrit.edu.in, nikhilvarmanamburi@gmail.com, bindusurisetty19@gmail.com, snamrata38@gmail.com nivedithanive62@gmail.com.

Abstract
The aim of this project is to solve the problem of modeling video behavior recorded in surveillance videos for use in online normal behavior recognition and anomaly detection applications. Without any manual marking of the training data collection, a new architecture is created for automated behavior profiling and online anomaly sampling/detection. The following are the core components of the framework: Based on discrete-scene event detection, a compact and efficient behavior representation method is developed. Modeling each pattern using a Dynamic Bayesian Network is used to evaluate the similarities between behavior patterns (DBN). A novel spectral clustering algorithm based on unsupervised model selection and feature selection on the eigenvectors of a normalized affinity matrix is used to discover the natural grouping of behavior patterns. To detect abnormal behavior, a runtime accumulative anomaly measure is implemented, while normal behavior patterns are recognized when adequate visual evidence is available based on an online survey. This allows the fastest possible identification and recognition of abnormalities and normal behavior. Experiments with noisy and fragmented data sets gathered from both indoor and outdoor surveillance scenarios show the efficacy and robustness of our approach. It is demonstrated that in detecting anomaly from an unseen video, a behavior model trained with an unlabeled data set outperforms those trained with the same but labeled data set.

Keywords: Event Detection, Dynamic Bayesian Network, Unsupervised model

1. Introduction

The monitoring or observing the activity of a person in crowded scenes on the road, stations, airport and shops through the camera system is called ‘Surveillance’. These camera systems mounted in and around different places and locations are used to capture the suspicious activity of any person and to avoid the forth coming dangers and threats to the common public. The recorded videos from these camera systems also help to catch the particular person or group of people who are responsible for the crime on the scene.

Cite this article as: K Madhavi, N.Nikhil Varma, S.Bindu Madhavi, Namrata Singh, E.Niveditha, "Abnormal Event Detection with CNN and Unsupervised Classification", International Journal of Research in Advanced Computer Science Engineering, (IJRACSE), Volume 7 Issue 2, July 2021, Page 1-9.

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

The outdoor surveillance system utilizes visible cameras for various other applications as well like for pedestrian detection, face recognition and human pose estimation in the domain of computer vision technology.

Abnormal event detection is one such application of surveillance system which has been used as an extensive research topic for several years. There are various others event detection processes like motion detection and person identification. Abnormality detection is one such sub-topic where the events are classified in to normal and abnormal respectively. In this thesis, the literature related to abnormal event detection can be classified into two categories, one is literature where abnormality detection has been perform educing hand crafted features and another onec on tains the literature where abnormality detection has been obtained using deep learning features. First, the literature related to handcrafted feature methods will be described and then the deep learning related literature are vividly analyzed.

With the ubiquitous deployment of surveillance cameras in public and private places, the Reis always a huge demand for utilizing those cameras for various applications such as crowd monitoring and abnormal activity detection. Due to the affordability of visible cameras, they are commonly installed in places where monitoring is required such as airports, railway and bus stations, shopping complexes and educational institutions. Further more, video surveillance techniques are also widely used in private places like within houses and child care centers to monitor different activities performed by elderly people and children in order to ensure their safety. These cameras mounted in and around different places and locations help in detecting suspicious behavior and avoiding forth coming dangers and threats to the common public. Visible cameras are also used in outdoor surveillance systems for applications like pede strian detection, face recognition and human pose estimation. They are also widely used for abnormality detection and therefore, several datasets for different detection purposes have been created. During this project, experiments were performed using the UCSD dataset due to two main reasons; first, the images within this data set have an optimum size for training purpose, and its popular use in enhancing surveillance capabilities within educational institutes.

The images captured from a camera in the UCSD dataset provides information about the pedestrians walking towards and away from the camera, while a second camera placed orthogonal to the first camera also collects the side view information. Therefore, the detection of abnormal events captured using the two cameras separately posesa challenge for both frame-based and patch-based detections.

Several handcrafted features have been utilized in the past to detect abnormalities. The recent evolution of machine learning techniques in various computer vision applications and the advancement in big data analysis and GPU computing provided an opportunity to implement a deep learning

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

algorithm for abnormality detection. Deep convolutional neural networks were not intensively explored in the past for surveillance purposes as they require abundant image data sets in order to achieve the desired outputs. Therefore, the aim of this project is to utilize visible camera images

2. System Analysis

Term system is derived from the Greek word 'systema' which means an organized relationship among functioning units or components. A system is an orderly grouping of interdependent components linked together according to a plan to achieve a specific objective.

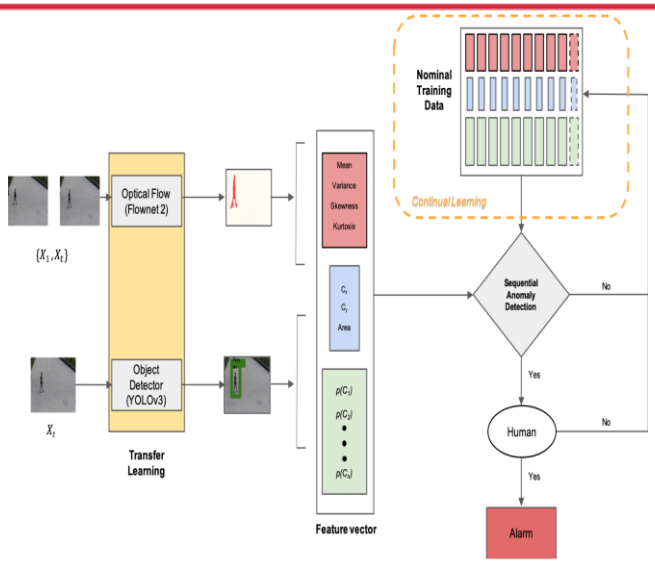
2.1 Existing System

Human activity identification has a wide range of uses. Recognizing a human activity may be useful in hospitals for monitoring patients and in public areas for providing protection and security. The installation of CCTV (Close Circuit Television) cameras has become mandatory due to the rise in various types of crime rates. However, manpower is needed to track the surveillance cameras. As a result, extensive research has been undertaken to simplify the monitoring process and detect suspicious human behavior. The Viola Jones algorithm is used to detect and recognize objects, which in this case is the human face, as stated in the research paper [1.] The Viola Jones algorithm can be used to detect and classify items, such as a human face, in an examination hall to reduce invigilator workload. Which is the human face in this case. The integral image is used to represent the features, which are then computed using pixel-based operations.

2.2 Proposed System

The definition of a simple deep neural network model for anomaly detection has been widely adopted in this work DNN is a mathematical model that analyses a large number of classified video input frames in general terms. Neural network architecture is based on the basic nerve system found in animal brains. CNN's Network, is amulti-perceptron network that is entirely connected. Weights and biases are called network neurons in the critical network. Dot product and neuron weight are two of the network's key processes. As shown in Figure 2, the Convolutional Neural Network has two important layers: pooling and completely linked layers. The convolutional network's key function is performed by the CNN network layers. At the same time, each neuron in the network layer is linked to a small region of nearby frame data. The receptive field refers to each person small area. The CNN model detects anomalies in video events in the proposed scheme. Frames from images are used to derive anomaly case characteristics. YouTube videos have been added to the latest dataset. For anomaly detection, the features are averaged from video frames and fed into a regular CNN classifier. Anomaly features from the layers of the model structured with different kernels for anomaly detection are used to train the CNN model. Using a feed forward process, the feature representations.

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com



3. System Requirements

SOFTWARE REQUIREMENTS

1.Operating system: - Windows 10
2.Coding Language: - python 3.6.4
3.IDE: - Jupiter notebook, GOOGLE COLAB
4.Dataset: - LV dataset
5.Video & Audio convertor: – ffmpeg
6.Visual C++ 2015 redistribution

HARDWARE REQUIREMENTS

1.Processor: - Core i5 or higher
2.Speed: - 2.1Ghz
3.RAM: - 8GB (min)
4.Hard Disk: - 200GB
5.GPU: - Integrated

Tools used:

Anaconda, Jupyter, Pycharm, etc.

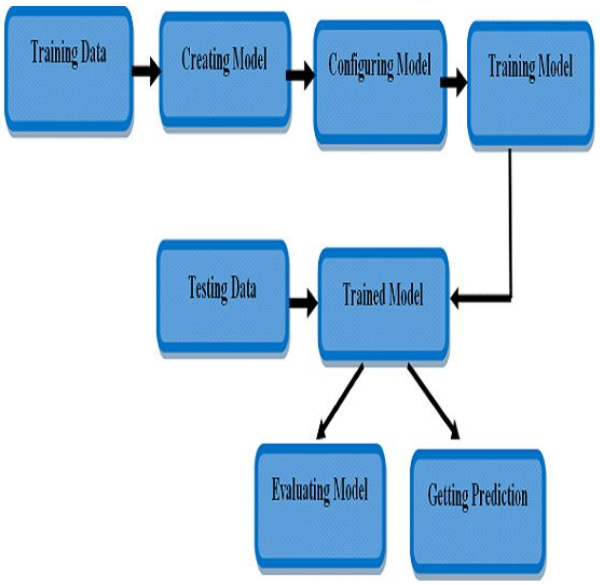
Languagesused:

R, Python, Matlab, CPP, Java, Julia, Lisp, Java Script, etc.

3.1 SYSTEM DESIGN

Architecture

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in way that supports reasoning about the structures and behaviors of the system.



4. Development

Software development is the process of conceiving, specifying, designing, programming, documenting, testing, and bug fixing involved in creating and maintaining applications, frameworks, or other software components.

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

FINAL CODES

```
from keras.preprocessing.image import img_to_array,load_img
import numpy as np
import glob
import os
from keras.layers import Conv3D,ConvLSTM2D,Conv3DTranspose
from keras.models import Sequential
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint
import imutils
import cv2

#Initialize directory path variable and describe a function to process and store video frames:
store_image=[]
train_path='train'
fps=5
trian_videos=os.listdir(train_path)
train_images_path=train_path+'/frames'
framepath=train_path+'/frames'

def store_inarray(image_path):
        image=load_img(image_path)
        image=img_to_array(image)
        image=cv2.resize(image, (227,227), interpolation = cv2.INTER_AREA)
        gray=0.2989*image[:,:,0]+0.5870*image[:,:,1]+0.1140*image[:,:,2]
        store_image.append(gray)

#Extract frames from video and call store function:
for video in trian_videos:
        os.system( 'ffmpeg -i {}/{} -r 1/1  {}/frames/%03d.jpg'.format(train_path,video,fps,train_path))
        images=os.listdir(train_images_path)
        for image in images:
            image_path=framepath+ '/'+ image
            store_inarray(image_path)
#Store the store_image list in a numpy file "training.npy":
store_image=np.array(store_image)
a,b,c=store_image.shape
store_image.resize(b,c,a)
store_image=(store_image-store_image.mean())/(store_image.std())
store_image=np.clip(store_image,0,1)
np.save('training.npy',store_image)
```

TRAIN.PY

```
#Create spatial autoencoder architecture:

stae_model=Sequential()
stae_model.add(Conv3D(filters=128,kernel_size=(11,11,1),strides=(4,4,1),padding="valid",input_shape=(227,227,10,1),activation='tanh'))
stae_model.add(Conv3D(filters=64,kernel_size=(5,5,1),strides=(2,2,1),padding="valid",activation='tanh'))
stae_model.add(ConvLSTM2D(filters=64,kernel_size=(3,3),strides=1,padding='same',dropout=0.4,recurrent_dropout=0.3,return_sequences=True))
stae_model.add(ConvLSTM2D(filters=32,kernel_size=(3,3),strides=1,padding='same',dropout=0.3,return_sequences=True))
stae_model.add(ConvLSTM2D(filters=64,kernel_size=(3,3),strides=1,return_sequences=True, padding='same',dropout=0.5))
stae_model.add(Conv3DTranspose(filters=128,kernel_size=(5,5,1),strides=(2,2,1),padding='valid',activation='tanh'))
stae_model.add(Conv3DTranspose(filters=1,kernel_size=(11,11,1),strides=(4,4,1),padding='valid',activation='tanh'))

stae_model.compile(optimizer='adam',loss='mean_squared_error',metrics=['accuracy'])

#Train the autoencoder on the "training.npy" file and save the model with name "saved_model.h5":

training_data=np.load('training.npy')
frames=training_data.shape[2]
frames=frames-frames%10

training_data=training_data[:,:,:frames]
training_data=training_data.reshape(-1,227,227,10)
training_data=np.expand_dims(training_data,axis=4)
target_data=training_data.copy()

epochs=5
batch_size=1

callback_save = ModelCheckpoint("saved_model.h5",monitor="mean_squared_error", save_best_only=True)

callback_early_stopping = EarlyStopping(monitor='val_loss', patience=3)


stae_model.fit(training_data,target_data,
        batch_size=batch_size,
        epochs=epochs,
        callbacks = [callback_save,callback_early_stopping]
        )
stae_model.save("saved_model.h5")
```

Test. PY

```
import cv2
import numpy as np
from keras.models import load_model
import argparse
from PIL import Image
import imutils


def mean_squared_loss(x1,x2):
    difference=x1-x2
    a,b,c,d,e=difference.shape
    n_samples=a*b*c*d*e
    sq_difference=difference**2
    Sum=sq_difference.sum()
    distance=np.sqrt(Sum)
    mean_distance=distance/n_samples

    return mean_distance

model=load_model("saved_model.h5")

cap = cv2.VideoCapture("crash5.mp4")
print(cap.isOpened())
print("requset processing")

while cap.isOpened():
    imagedump=[]
    ret,frame=cap.read()


    for i in range(10):
        ret,frame=cap.read()
        image = cv2.resize(frame,(1000, 1200))
        frame=cv2.resize(frame, (227,227), interpolation = cv2.INTER_AREA)
        gray=0.2989*frame[:,:,0]+0.5870*frame[:,:,1]+0.1140*frame[:,:,2]
        gray=(gray-gray.mean())/gray.std()
        gray=np.clip(gray,0,1)
        imagedump.append(gray)

    imagedump=np.array(imagedump)

    imagedump.resize(227,227,10)
    imagedump=np.expand_dims(imagedump,axis=0)
    imagedump=np.expand_dims(imagedump,axis=4)
    print("checking")

    output=model.predict(imagedump)

    loss=mean_squared_loss(imagedump,output)

    if frame.any()==None:
        print("none")


    if cv2.waitKey(10) & 0xFF==ord('q'):
        break

    test_list = [0.0003737973240346733,0.0003738015888208667 3,0.0003705208840195235,0.0003596726936734095
    x = len(test_list)
    for i in range(x):
        if loss == test_list[i]:
            print('Abnormal Event Detected')
            cv2.putText(image,"Abnormal Event",(220,100),cv2.FONT_HERSHEY_SIMPLEX,2,(0,0,255),4)
            if i == x-1:
                import mail_test
    cv2.imshow("video",image)

cap.release()
cv2.destroyAllWindows()
```

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

5. System Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

5.1 Types of Tests

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.
Invalid Input: identified classes of invalid input must be rejected.
Functions: identified functions must be exercised.
Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

testing is complete, additional tests are identified and the effective value of current tests is determined.

5.2 System Testing

Testing Methodologies

The following are the Testing Methodologies:

- Unit Testing.
- Integration Testing.
- User Acceptance Testing.
- Output Testing.
- Validation Testing.

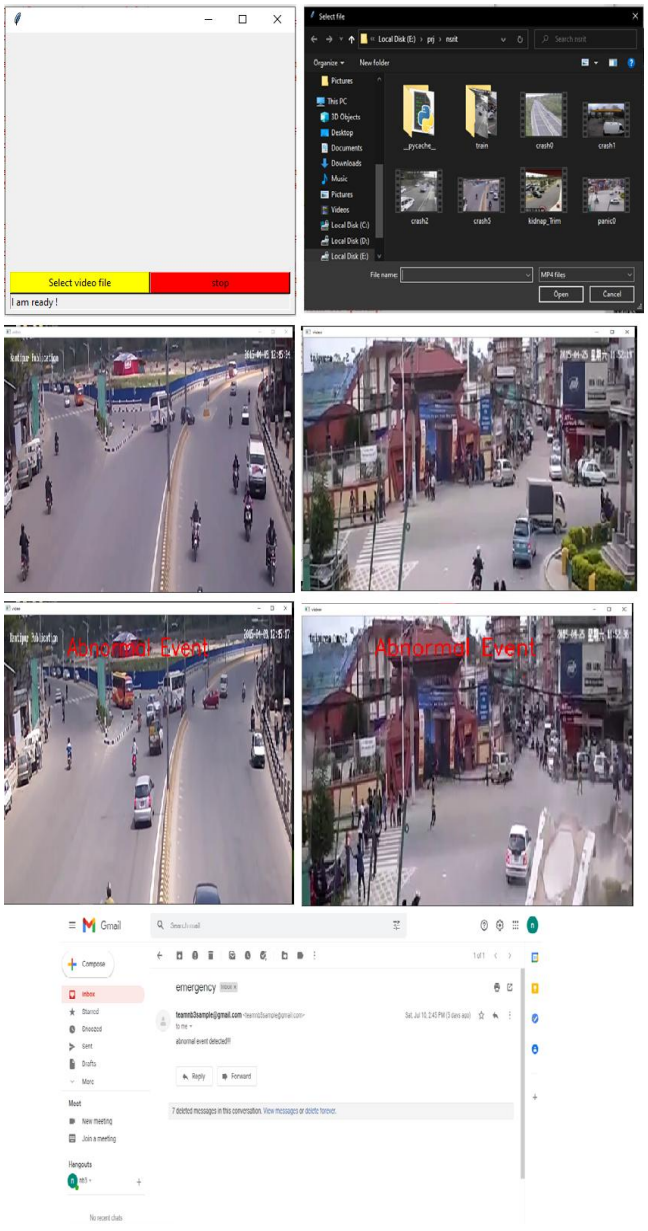
6. Results

6.1 Testing Strategy:

A strategy for system testing integrates system test cases and design techniques into a well-planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation. A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements. Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

6.2 System Testing:

Software once validated must be combined with other system elements (e.g., Hardware, people, database). System testing verifies that all the elements are proper and that overall system function performance is achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation.



ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

Conclusions

Using deep learning methods, the proposed method detects anomalies. The device generates voice alerts and alarms in three different ways after detecting the anomaly zone. The approach is effective in identifying the source of the fire. The proposed system can be used in crowded areas with a large number of people. Overall, the precision is greater than 90%. The method may be used to introduce more different form so fan omalies in the future.

References

1. Popoola, O.P. and K.Wang, Video-based abnormal human behavior recognition—Are view. IEEE Transactions on Systems, Man, and Cybernetics, PartC (Applications and Reviews), 2012.42(6): p.865-878.
2. Jiang,F., Yuan,J., Tsaftaris, S.A., & Katsaggelos, A.K., Anomalous video event detection using spatiotemporal context. Computer Vision and Image Understanding,2011.115(3): p. 323-333.
3. Kim, J. and K. Grauman., Observe locally, infer globally: a space-time RF for detecting abnormal activities with incremental updates.in Computer Vision and Pattern Recognition, 2009. CVPR2009. IEEE Conference on. 2009. IEEE.
4. Kratz, L. and K. Nishino., Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models.in Computer Vision and Pattern Recognition, 2009. CVPR2009. IEEE Conference on. 2009.IEEE.
5. Xu, J., Denman, S., Fookes, C., & Sridharan, S., Unusual event detection in crowded scenes using bag of LBPs in spatio-temporal patches. in Digital Image Computing Techniques and Applications (DICTA), 2011 International Conference on. 2011.IEEE.
6. Nallaivarothayan, H., Fookes, C., Denman, S., & Sridharan, S., An MRF based abnormal event detection approach using motion and appearance features. In Advanced Video and Signal Based Surveillance(AVSS), 2014 11th IEEE International Conference on. 2014.IEEE.
7. Ryan, D., Denman, S., Fookes, C., & Sridharan, S., Textures of optical flow for real-time anomaly detection in crowds. in Advanced Video and Signal-Based Surveillance(AVSS), 2011 8th IEEE International Conference on.2011.IEEE.
8. Jiang, F., Y. Wu, and A.K. Katsaggelos., Abnormal event detection from surveillance video by dynamic hierarchical clustering. in Image Processing,2007. ICIP2007. IEEE International Conference on. 2007.IEEE.
9. Lee, C.-K., Ho, M. F., Wen, W. S., & Huang, C. L., Abnormal event detection in video using n-cutclustering .in Intelligent Information Hiding and Multimedia Signal Processing, 2006. IIH-MSP'06. International Conference on. 2006.IEEE.
10. Wang, Y.,K.Huang, and T.Tan., Abnormal activity recognition in office based on R transform. in Image Processing, 2007. ICIP 2007. IEEE International Conference on. 2007.IEEE.
11. Li,J.,Hopedale's, T.M.,Gong,S.,& Xiang, T.,Learningr are behaviors.in Asia Conference on Computer Vision. 2010. Springer.

ISSN No : 2454-4221 (Print)
ISSN No : 2454-423X (Online)
International Journal of Research in Advanced Computer Science Engineering
A Peer Reviewed Open Access International Journal
www.ijracse.com

12. Ionescu, R.T.,Smeureanu,S., Alexe,B., & Popescu, M.,Unmasking the abnormal events in video. arXiv preprint arXiv:1705.08182, 2017.