

HUMAN ABNORMAL ACTIVITY RECOGNITION

M V Sowdhamini Bharadwaj¹, N Harshini², P Mounika³, Mrs. M Sowmya⁴

Department of Computer Science and Engineering, Stanley College of Engineering and Technology for Women, Telangana, India

Abstract. This paper brings forward one amongst the foremost significant applications of human suspicious activity recognition that is termed as anomaly detection. A key concern of any society today is providing safety to an individual. The main reason behind this concern is due to the constantly increasing activities causing threats, starting from deliberate ferocity to an injury caused through an accident. Simple installation of a traditional closed circuit television(CCTV) is not sufficient as it requires a person to continuously stay alert and monitor the cameras, which is quite inefficient. This call for the requirement to develop an security system which is fully automated system that recognizes anomalous activities in real time and brings instant help to the victims. Hence we proposed a system which will examine and detect the suspicious human action from real-time CCTV footage with help of machine learning techniques and generates the alert if the abnormal activity is occurred. The method is implemented on the dataset containing both normal and anomaly activity and experiment has shown better results.

1. Introduction

1.1 About Project

Human abnormal activity recognition can be useful to a variety of scenarios, and anomaly detection in security systems is one of among them. Seen the increasing demand for security, surveillance cameras have been widely set up as the infrastructure for image analysis. One of the major challenges faced by surveillance image analysis is detecting abnormal activity which requires exhausting human efforts. Fortunately, such a labor-intensive task can be recast as an anomaly detection problem which aims to detect unexpected actions or patterns. Anomaly detection varies from the traditional classification problem in the following aspects:

- 1)It is very difficult to list all possible negative (anomaly) illustrations.
- 2) It is a daunting job to collect adequate negative samples due to the rarity. An activity recognition system is projected to identify the basic day to day activities performed by a human being. It is challenging to achieve high rate accuracy for recognition of these activities due to the complexity and diversity in human activities. Activity models required for identification and classification of human activities are constructed based on different approaches specific to the application. The activities of a human being can be generally categorized into normal activities or anomalous activities. A human being's deviation from normal behavior to abnormal causing harm to the surrounding or to himself is classified as an anomalous activity. To achieve anomaly detection, one of the most widespread method

is using the images of abnormal events as training data to learn a model and then detecting the suspicious events which would fit in the learned model.

1.2 Objectives of the Project

The main objectives identified which illustrate the relevance of the topic are listed out below.

To prepare a dataset consisting of images for 3 activities namely Kick, Punch, Push, Hit. The dataset is segmented and the relevant data is annotated with labels of the abovementioned activities. Data is pre-processed and redundant and irrelevant data is edited out. The development of an optimised model is carried out that uses CNN as a classifier to segment the dataset images into their respective action labels. This is done following a supervised learning procedure for training the model. Demonstrating the results.

1.3 Scope of the Project

Human Activity Recognition can benefit various applications in fields like smart home monitoring, healthcare services, security surveillance, childcare etc. In future we can update this application by using object activity recognition in which activities performed by objects can also be tracked and analyzed. Application of integrated large datasets can be done to identify the activity taking place as slower rate of time. Even very subtle or minute variations should be recognized by the system. The data of actor performing the anomalous activity can be stored and identification of actor can be done if not caught in the first place. Activities that are of reoccurring manner should be stored to save time and space during recognition process. Implementation of such model can also be done in Government authority section. Much more developments for improvisation in accuracy and dealing with issues related to optical identity and background clutter of image can be done.

2. Literature Survey

2.1 Existing System

The existing system was manual where a person had to sit in front of a mirror to monitor and guide human activities, it was hectic, time consuming and costlier system and was prone to human errors and negligence. Further some systems started using sensor data to recognize human activities but they were needed to be worn by the user which limited the scope of activity recognition in open environment in general.

2.2 Proposed System

In our proposed system, for detecting anomalous behavior, the CNN i.e. convolution neural network have been used. For effectively classification of anomalous activities, it is essential to recognize the temporal data in the image. Recently, CNN is mostly used for extracting key features from each frame of the image. CNN is only the algorithm best suited for this purpose. For classifying the given input successful, it is necessary that the features get extracted from CNN, therefore CNN should be capable of knowing and extracting the needed features from the frame of image.

3. Proposed Architecture

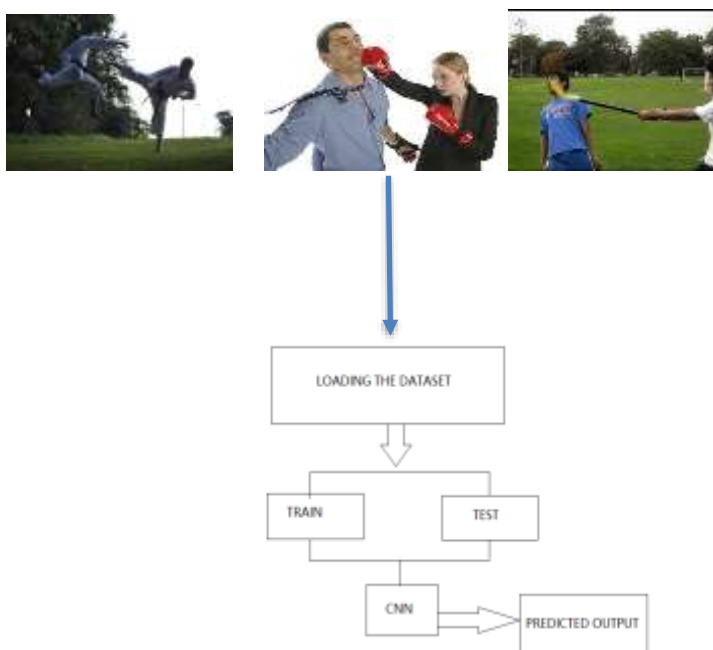


Fig.1. Proposed Architecture

4. Implementation

4.1 Algorithm

The various deep learning methods use data to train neural network algorithms to do a variety of machine learning tasks, such as the classification of different classes of objects. Convolutional neural networks are deep learning algorithms that are very powerful for the analysis of images. This article will explain to you how to construct, train and evaluate convolutional neural networks.

There are three types of layers in Convolutional Neural Networks:

- 1) Convolutional Layer: In a typical neural network each input neuron is connected to the next hidden layer. In CNN, only a small region of the input layer neurons connect to the neuron hidden layer.
- 2) Pooling Layer: The pooling layer is used to reduce the dimensionality of the feature map. There will be multiple activation & pooling layers inside the hidden layer of the CNN.
- 3) Fully-Connected layer: Fully Connected Layers form the last few layers in the network. The input to the fully connected layer is the output from the final Pooling or Convolutional Layer, which is flattened and then fed into the fully connected layer.

Step-1: Import key libraries

Step-2: Reshape the data

Each image is 28X28 size, so there are 784 pixels. So, the output layer has 10 outputs, the hidden layer has 784 neurons and the input layer has 784 inputs. The dataset is then converted into float datatype.

Step-3: Normalize the data

NN models usually require scaled data. In this code snippet, the data is normalized from (0-255) to (0-1) and the target variable is one-hot encoded for further analysis. The target variable has a total of 10 classes (0-9)

Step-4: Define the model function

There are two layers one is a hidden layer with activation function ReLu and the other one is the output layer using the softmax function.

Step-5: Run the model

4.2 Code Implementation

Anaconda Navigator Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda® distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda.org or in a local Anaconda Repository. It is available for Windows, macOS, and Linux. In order to run, many scientific packages depend on specific versions of other packages. Data scientists often use multiple versions of many packages and use multiple environments to separate these different versions. The command-line program conda is both a package manager and an environment manager. This helps data scientists ensure that each version of

each package has all the dependencies it requires and works correctly. Navigator is an easy, point-and-click way to work with packages and environments without needing to type conda commands in a terminal window. You can use it to find the packages you want, install them in an environment, run the packages, and update them – all inside Navigator.

Anaconda Prompt Anaconda command prompt is just like command prompt, but it makes sure that you are able to use anaconda and conda commands from the prompt, without having to change directories or your path. When you start Anaconda command prompt, you'll notice that it adds/("prepends") a bunch of locations to your PATH.

Python 3.7. Python is broadly utilized universally and is a high-level programming language. It was primarily introduced for prominence on code, and its language structure enables software engineers to express ideas in fewer lines of code. Python is a programming language that gives you a chance to work rapidly and coordinate frameworks more effectively.

Jupyter Notebook Jupyter Notebook is an open-source, web-based interactive environment, which allows you to create and share documents that contain live code, mathematical equations, graphics, maps, plots, visualizations, and narrative text. It integrates with many programming languages like Python, PHP, R, C#, etc.

1. All in one place: As you know, Jupyter Notebook is an open-source web-based interactive environment that combines code, text, images, videos, mathematical equations, plots, maps, graphical user interface and widgets to a single document.
2. Easy to convert: Jupyter Notebook allows users to convert the notebooks into other formats such as HTML and PDF. It also uses online tools and nbviewer which allows you to render a publicly available notebook in the browser directly.
3. Easy to share: Jupyter Notebooks are saved in the structured text files (JSON format), which makes them easily shareable.
4. Language independent: Jupyter Notebook is platform-independent because it is represented as JSON (JavaScript Object Notation) format, which is a language-independent, text-based file format. Another reason is that the notebook can be processed by any programming language, and can be converted to any file formats such as Markdown, HTML, PDF, and others.
5. Interactive code: Jupyter notebook uses ipywidgets packages, which provide many common user interfaces for exploring code and data interactivity.

5. Result

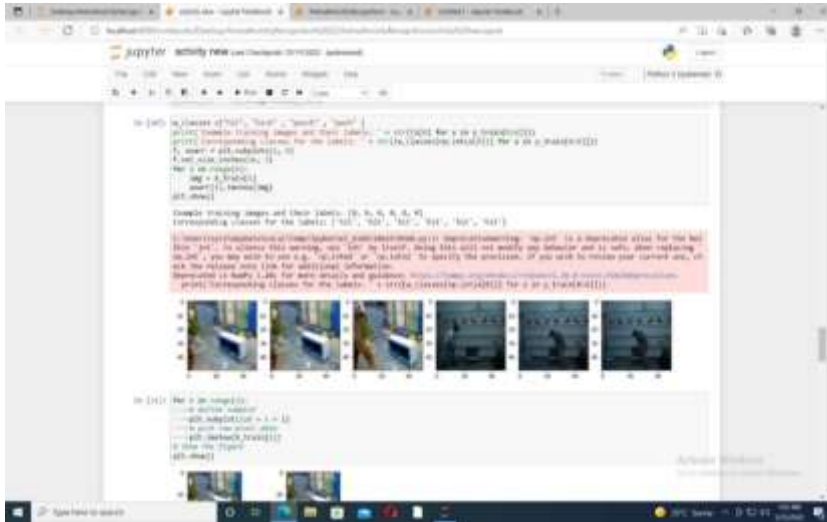


Fig 5.1 Plotting

#Training and Testing

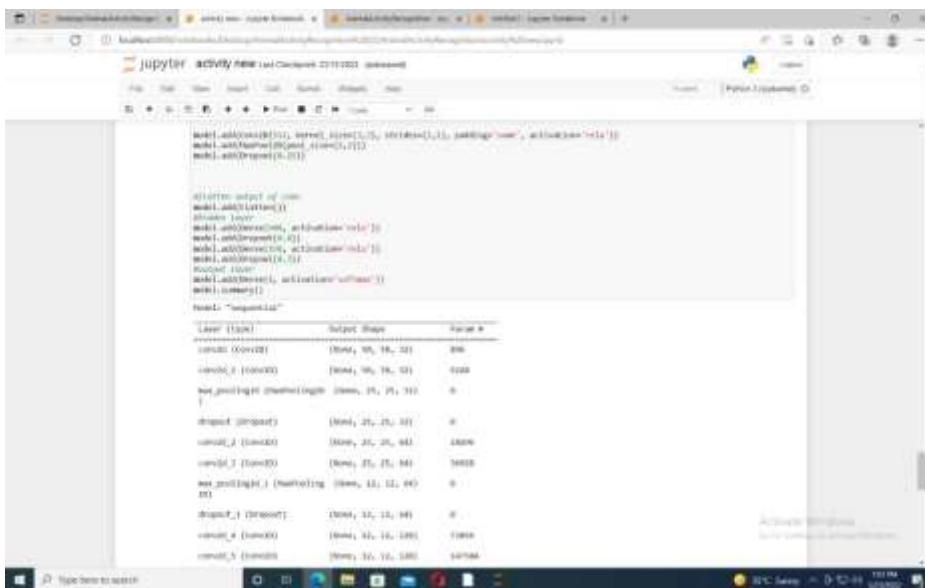
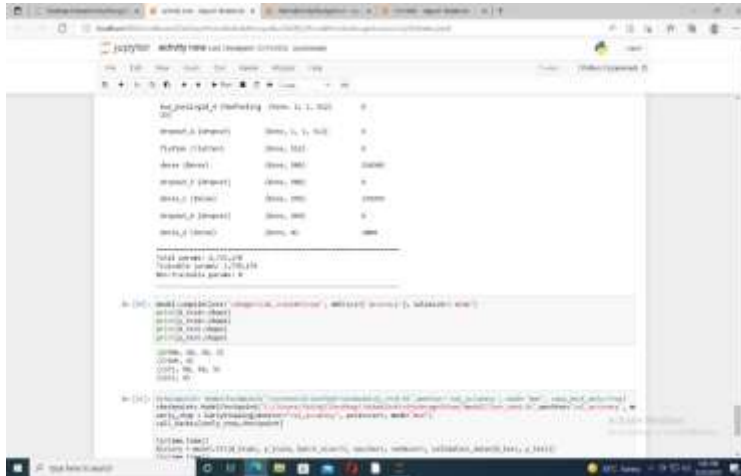


Fig 5.2



```

x_train_scaled_4 (overfitting) (shape: (4, 1, 512)) 0
dropout_4 (Dropout) (shape: (1, 1, 512)) 0
flatten_4 (Flatten) (shape: (512)) 0
dense_4 (Dense) (shape: (512)) 25088
dropout_5 (Dropout) (shape: (512)) 0
dense_5 (Dense) (shape: (256)) 12288
dropout_6 (Dropout) (shape: (256)) 0
dense_6 (Dense) (shape: (4)) 1684

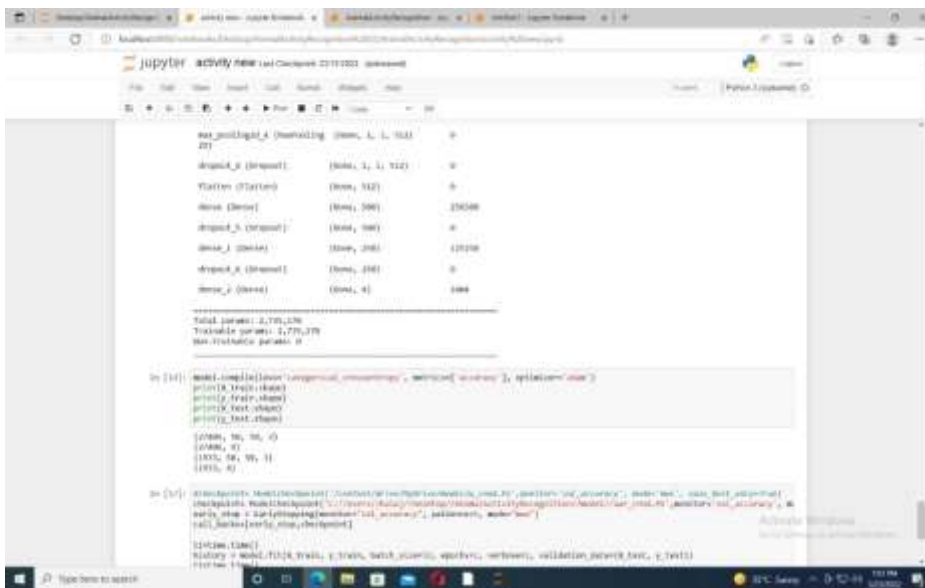
Total params: 4,779,236
Trainable params: 1,250,816
Non-trainable params: 0

In [10]: model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
          print(model.summary())
          print(model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'], validation_data=(x_val_scaled_4, y_val_scaled_4)))
          model.fit(x_train_scaled_4, y_train_scaled_4, validation_data=(x_val_scaled_4, y_val_scaled_4), epochs=10)

Epoch 0/10:
1/10 [====] - loss: 0.6921 - accuracy: 0.1250
2/10 [====] - loss: 0.6921 - accuracy: 0.1250
3/10 [====] - loss: 0.6921 - accuracy: 0.1250
4/10 [====] - loss: 0.6921 - accuracy: 0.1250
5/10 [====] - loss: 0.6921 - accuracy: 0.1250
6/10 [====] - loss: 0.6921 - accuracy: 0.1250
7/10 [====] - loss: 0.6921 - accuracy: 0.1250
8/10 [====] - loss: 0.6921 - accuracy: 0.1250
9/10 [====] - loss: 0.6921 - accuracy: 0.1250
10/10 [====] - loss: 0.6921 - accuracy: 0.1250
10 epochs: 1.000s
History: <History object>
  
```

Fig 5.3

Fig 5.2 & Fig 5.3 - Testing and Training the model



```

x_train_scaled_4 (overfitting) (shape: (4, 1, 512)) 0
dropout_4 (Dropout) (shape: (1, 1, 512)) 0
flatten_4 (Flatten) (shape: (512)) 0
dense_4 (Dense) (shape: (512)) 25088
dropout_5 (Dropout) (shape: (512)) 0
dense_5 (Dense) (shape: (256)) 12288
dropout_6 (Dropout) (shape: (256)) 0
dense_6 (Dense) (shape: (4)) 1684

Total params: 4,779,236
Trainable params: 1,250,816
Non-trainable params: 0

In [11]: model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
          print(model.summary())
          print(model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'], validation_data=(x_val_scaled_4, y_val_scaled_4)))
          model.fit(x_train_scaled_4, y_train_scaled_4, validation_data=(x_val_scaled_4, y_val_scaled_4), epochs=10)

Epoch 0/10:
1/10 [====] - loss: 0.6921 - accuracy: 0.1250
2/10 [====] - loss: 0.6921 - accuracy: 0.1250
3/10 [====] - loss: 0.6921 - accuracy: 0.1250
4/10 [====] - loss: 0.6921 - accuracy: 0.1250
5/10 [====] - loss: 0.6921 - accuracy: 0.1250
6/10 [====] - loss: 0.6921 - accuracy: 0.1250
7/10 [====] - loss: 0.6921 - accuracy: 0.1250
8/10 [====] - loss: 0.6921 - accuracy: 0.1250
9/10 [====] - loss: 0.6921 - accuracy: 0.1250
10/10 [====] - loss: 0.6921 - accuracy: 0.1250
10 epochs: 1.000s
History: <History object>
  
```

Fig 5.4 Predicting the accuracy

6. Conclusion

Human activity understanding has become one of the most active research topics in computer vision. The type and amount of data that each approach uses depends on the ability of the underlying algorithm to deal with heterogeneous and/or large scale data. The development of a fully automated human activity recognition system is a non-trivial task due to cluttered backgrounds, complex camera motion, large intraclass variations, and data acquisition issues.

Human activity recognition remains to be an important problem in computer vision. HAR is the basis for many applications such as video surveillance, health care, and human-computer interaction. Methodologies and technologies have made tremendous development in the past decades and have kept developing up to date. However, challenges still exist when facing realistic sceneries, in addition to the inherent intraclass variation and interclass similarity problem.

7. Future Scope

Since computer vision is a trending topic in these days, systems like Human Activity Recognition systems is quite useful and effective for solving a variety of application, whether it would be surveillance or monitoring, or aiding the elderly and blind people etc. This not only provide additional comfort to the end-users but also can be deployed into different Organizations in order to reduce the employ workload. The model shows good results on video streams while performing decently on image data. Activity Recognition system are of great importance in modern days due to the convenience and problems which the system offers and solves. Need of Activity recognition for monitoring and surveillance, video segmentation etc is of growing demand in which this system can greatly help. This system can be incorporated in mobiles apps to further aid the elderly and blind people. It is cost-effective and an immense time saving system which is also prone to human errors. This system acts as a base solution for many other applications involving activity recognition. Hence, this system is very beneficial for both individual and organizations for general or specialize purposes.

7. References

- [1] <https://www.computerscijournal.org/vol13no23/human-activity-recognition-system/#:~:text=Conclusion%20and%20Future%20Scope&text=Activity%20Recognition%20system%20are%20of,this%20system%20can%20greatly%20help>.
- [2] https://www.researchgate.net/publication/337664519_Human_activity_recognition_by_using_convolutional_neural_network
- [3] <https://www.frontiersin.org/articles/10.3389/frobt.2015.00028/full>
- [4] <https://satwikkottur.github.io/reports/HumanActivity-Report.pdf>

5. [5] <https://www.geeksforgeeks.org/human-activity-recognition-using-deep-learning-model/>
6. [6] <https://thecleverprogrammer.com/2021/01/10/human-activity-recognition-with-machine-learning/>
7. [7] <https://arxiv.org/abs/2202.03274>
8. [8] <https://link.springer.com/article/10.1007/s42486-020-00026-2#:~:text=The%20use%20of%20Convolutional%20Neural,CNNs%20can%20extract%20features%20a,automatically>
9. Kishor Kumar Reddy C and Vijaya Babu B, “ISPM: Improved Snow Prediction Model to Nowcast the Presence of Snow/No-Snow”, International Review on Computers and Software, 2015.
10. (<http://www.praiseworthyprize.org/jsm/index.php?journal=irecos&page=article&op=view&path%5B%5D=17055>)
11. Kishor Kumar Reddy C, Rupa C H and Vijaya Babu B, “SLGAS: Supervised Learning using Gain Ratio as Attribute Selection Measure to Nowcast Snow/No-Snow”, International Review on Computers and Software, 2015.
12. (<http://www.praiseworthyprize.org/jsm/index.php?journal=irecos&page=article&op=view&path%5B%5D=16706>)
13. Kishor Kumar Reddy C, Vijaya Babu B, Rupa C H, “SLEAS: Supervised Learning using Entropy as Attribute Selection Measure”, International Journal of Engineering and Technology, 2014.
14. (<http://www.enggjournals.com/ijet/docs/IJET14-06-05-210.pdf>)
15. Kishor Kumar Reddy C, Rupa C H and Vijaya Babu B, “A Pragmatic Methodology to Predict the Presence of Snow/No-Snow using Supervised Learning Methodologies”, International Journal of Applied Engineering Research, 2014.
16. (<http://www.ripublication.com/Volume/ijaerv9n21.htm>)
17. Kishor Kumar Reddy C, Rupa C H and Vijaya Babu, “SPM: A Fast and Scalable Model for Predicting Snow/No-Snow”, World Applied Sciences Journal, 2014