**Intelligent Video Surveillance Using Deep Learning**

**1. INTRODUCTION**

During these recent years, applications of video surveillance have attracted more and more researchers. Consequently, various types of modelling, as well as several techniques of analysis and detection of human activities, are suggested. Particularly, many pieces of research are involved in the recognition and detection of human activities in general and especially abnormal activities. One important application is the supervision of elderly and disabled people at home in care centers, or hospitals. Recognition of human activities is a recent field that is interested to provide techniques and methods allowing the detection and classification of human activities, and extended now to recognize normal or abnormal activities. The motivation behind the latter is to provide an immediate intervention to preserve the lives of individuals or to ensure them some services they are unable to do by themselves. Being recent and interesting, this field has attracted the attention of several researchers who try to find solutions to the problems faced in studying such types of activities. However, the proposals made for this until now are those used for the recognition of normal human activities with minor modifications. These proposals are still very restricted because of the very limited number of works and surveys in this field. Moreover, they are not efficient and suffer from several limitations and technical difficulties. To this end, we propose in this paper an overview and an analysis of the existing works, to offer the researchers a general view of what exists in this field and to provide them with a tool being a help to them propose new approaches. For this, the manuscript is organized as follows. In the second section, we present a definition of the abnormal activities, their various types, as well as some examples of abnormal activities of a group or a single person. We then discuss in the third section the motivations that led to the advent of this research axis and the development of techniques allowing the analysis and recognition of human activities in general and abnormal activities in particular. The fourth section is devoted to the proposed approaches in the literature for the detection of abnormal activities. In this section, we present for each proposal, the purpose for which it is set up, its different stages, and the means used for its validation. Subsequently, we discuss some aspects affecting or influencing the effectiveness and credibility of the classification of human activities. The sixth section presents the three modes of automatic learning (supervised, unsupervised, and semisupervised). Thereafter, we enumerate the encountered limitations to be taken into consideration to improve the systems of recognition and identification of abnormal activities. Finally, we finish with a conclusion where we summarize our study.

**1.1 Objective of the Project**

Abnormal activity detection plays a very important role in surveillance applications. To capture the abnormal activity of humans without the intervention of the system i.e. automatically captures the video can be implemented. Human fall detection, suddenly jumping down which has an important application in the field of safety and security. Proposed system use for detecting roadside human activities or behaviour by using the Probabilistic Neural Network (PNN) method for classifying activities or behaviour between training dataset and testing videos. The partitions between classes of normal activities have also been learned using multi-PNNs. recognizing human activity has become a trend in smart surveillance that contains several challenges, such as performing effective detection of huge video data streams, while maintaining low computational complexity. Current activity recognition techniques are using convolutional neural network (CNN) model with computationally complex classifiers, creating hurdles in obtaining quick responses for abnormal activity, so this paper proposes a framework for activity detection. First, we detect abnormal activity with humans in the surveillance stream using an effective CNN model. The detected individual is tracked throughout the video stream via an ultrafast object tracker called ‘minimum output sum of squared error’ {MOSSE), Next, for each Tracked individual, pyramidal convolutional features are extracted from two consecutive frames using the efficient LiteFlowNet CNN. Finally, a novel deep skip connection gated recurrent unit is trained to learn different temporal changes in the sequence of frames for activity recognition and detection. We finish by the result indicate the efficiency of the proposed technique.

**2. LITERATURE SURVEY**

**A review of human activity recognition methods**

Recognizing human activities from video sequences or still images is a challenging task due to problems, such as background clutter, partial occlusion, changes in scale, viewpoint, lighting, and appearance. Many applications, including video surveillance systems, human-computer interaction, and robotics for human behaviour characterization, require a multiple activity recognition system. In this work, we provide a detailed review of recent and state-of-the-art research advances in the field of human activity classification. We propose a categorization of human activity methodologies and discuss their advantages and limitations. In particular, we divide human activity classification methods into two large categories according to whether they use data from different modalities or not. Then, each of these categories is further analyzed into sub-categories, which reflect how they model human activities and what type of activities they are interested in. Moreover, we provide a comprehensive analysis of the existing, publicly available human activity classification datasets and examine the requirements for an ideal human activity recognition dataset. Finally, we report the characteristics of future research directions and present some open issues on human activity recognition.

**Video-based abnormal human behaviour recognition—A review**

Modelling human behaviours and activity patterns for recognition or detection of special event has attracted significant research interest in recent years. Diverse methods that are abound for building intelligent vision systems aimed at scene understanding and making correct semantic inference from the observed dynamics of moving targets. Most applications are in surveillance, video content retrieval, and human-computer interfaces. This paper presents not only an update extending previous related surveys, but also a focus on contextual abnormal human behavior detection especially in video surveillance applications. The main purpose of this survey is to extensively identify existing methods and characterize the literature in a manner that brings key challenges to attention.

**Motion pattern extraction and event detection for automatic visual surveillance**

Efficient analysis of human behavior in video surveillance scenes is a very challenging problem. Most traditional approaches fail when applied in real conditions and contexts like amounts of persons, appearance ambiguity, and occlusion. In this work, we propose to deal with this problem by modeling the global motion information obtained from optical flow vectors. The obtained direction and magnitude models learn the dominant motion orientations and magnitudes at each spatial location of the scene and are used to detect the major motion patterns. The applied region-based segmentation algorithm groups local blocks that share the same motion direction and speed and allows a subregion of the scene to appear in different patterns. The second part of the approach consists in the detection of events related to groups of people which are merge, split, walk, run, local dispersion, and evacuation by analyzing the instantaneous optical flow vectors and comparing the learned models. The approach is validated and experimented on standard datasets of the computer vision community. The qualitative and quantitative results are discussed.

**Robust real-time unusual event detection using multiple fixed-location monitors**

We present a novel algorithm for detection of certain types of unusual events. The algorithm is based on multiple local monitors which collect low-level statistics. Each local monitor produces an alert if its current measurement is unusual, and these alerts are integrated to a final decision regarding the existence of an unusual event. Our algorithm satisfies a set of requirements that are critical for successful deployment of any large-scale surveillance system. In particular it requires a minimal setup (taking only a few minutes) and is fully automatic afterwards. Since it is not based on objects' tracks, it is robust and works well in crowded scenes where tracking-based algorithms are likely to fail. The algorithm is effective as soon as sufficeint low-level observations representing the routine activity have been collected, which usually happens after a few minutes. Our algorithm runs in realtime. It was tested on a variety of real-life crowded scenes. A ground-truth was extracted for these scenes, with respect to which detection and false-alarm rates are reported.

**Detection of abnormal living patterns for the elderly living alone using support vector data description**

In this study, we developed an automated behaviour analysis system using infrared (IR) motion sensors to assist the independent living of the elderly who live alone and to improve the efficiency of their healthcare. An IR motion-sensor-based activity-monitoring system was installed in the houses of the elderly subjects to collect motion signals and three different feature values, activity level, mobility level, and non-response interval (NRI). These factors were calculated from the measured motion signals. The support vector data description (SVDD) method was used to classify normal behaviour patterns and to detect abnormal behavioural patterns based on the aforementioned three feature values. The simulation data and real data were used to verify the proposed method in the individual analysis. A robust scheme is presented in this paper for optimally selecting the values of different parameters especially that of the scale parameter of the Gaussian kernel function involving in the training of the SVDD window length, T of the circadian rhythmic approach with the aim of applying the SVDD to the daily behaviour patterns calculated over 24 h. Accuracies by positive predictive value (PPV) were 95.8% and 90.5% for the simulation and real data, respectively. The results suggest that the monitoring system utilizing the IR motion sensors and abnormal-behaviour-pattern detection with SVDD are effective methods for home healthcare of elderly people living alone.

**Abnormal human activity recognition using SVM based approach**

Tele-health care applications have gained much attention in the field of ubiquitous computing. With availability of affordable wearable sensors, it is possible to recognize human activities. Activity detection has many applications such as security and health care applications. In this paper, the focus is on detecting abnormal activities of the individuals by ruling out all possible normal activities. Abnormal activities are unexpected events that occur in random manner. Human activities can be recognized using various approaches. Most widely used approach is multi-class SVM. This paper proposes a novel scheme of representing human activities in form of a state transition table. The transition table helps the classifier in avoiding the states which are unreachable from the current state. By avoiding the unreachable states, computational time for classification is reduced significantly when compared to conventional approaches. It is found from simulation studies that the proposed scheme gives accurate results with less computational complexity.

**Person-on-person violence detection in video data**

We address the problem of detecting human violence in video, such as fist fighting, kicking, hitting with objects, etc. To detect violence we rely on motion trajectory information and on orientation information of a person's limbs. We define an Acceleration Measure Vector (AMV) composed of direction and magnitude of motion and we define jerk to be the temporal derivative of AMV We present results from several data sequences involving multiple types of violent activities.

**Real-life violent social interaction detection**

This paper proposes a method to detect and localize dyadic human interactions in real videos. The idea stems from the significant difference between an action performed by a single subject and an interaction between two persons. In the first case all the visual information is concentrated on the subject, while in the latter case the action of a person is related to the interacting person's attitude, following an action/reaction principle. This kind of behavior is significant especially in natural and real scenarios, in which people are moving freely without the awareness of being recorded. To highlight these features and provide researchers with a common ground for comparisons, we have collected and annotated a new dataset, retrieving from YouTube 30 different videos of a specific type of interaction, namely urban fight situations. The proposed dataset is one of the most challenging annotated video collection concerning dyadic interactions, due to the intrinsic intra-class variability characterizing real fights. In addition, we provide an extensive experimental analysis on this dataset and we demonstrate that the visual information extracted in the area associated to the interpersonal space plays a fundamental role in detecting fights.

**Total Variation Regularization of Local-Global Optical Flow**

More data fidelity terms in variational optical flow methods improve the estimation's robustness. A robust and anisotropic smoother enhances the specific fill-in process. This work presents a combined local-global (CLG) approach with total variation regularization. The combination of bilateral filtering and anisotropic (image driven) regularization is used to control the propagation phenomena. The resulted method, CLG-TV, is able to compute larger displacements in a reasonable time. The numerical scheme is highly parallelizable and runs in real-time on current generation graphic processing units.

**Fight Detection in Surveillance Videos**

Fight detection is an important topic for surveillance systems. However, there has been little success in creating an algorithm that can detect fight in surveillance videos with high performance. In this work, we propose a new method for the task of fight detection in surveillance videos. The proposed method relies on a novel motion feature, namely Motion Co-Occurrence Feature (MCF). Firstly, motion vectors are extracted by using block matching algorithm. Secondly, direction and magnitude values of motion vectors are quantized separately. Afterwards, direction and magnitude based MCF is calculated by considering both current and past motion vectors. Experimental results obtained using k-Nearest Neighbour classifier showed that the proposed algorithm can discriminate fight scenes with significantly high accuracy.

**3. SYSTEM ANALYSIS**

**3.1 EXISTING SYSTEM**

This paper is much work on abnormal behaviour detection took a supervised learning approach. Diverse contributions have been made in the development of behaviour recognizers for smart building surveillance applications. In automatic roaders, human surveillance, the vehicle or human activities and behaviours are detected and recognized for monitoring and warning purposes, for detecting human behaviour. This technique only focuses on updating anomalous human activity detection. The hidden Markov Model (HMM) and Dynamic Bayesian Network Model (DBNM) are using to detect suspicious behaviour. Motion detection, tracking, and classification for automated video surveillance. In the existing system, the video surveillance system is designed for human operators to observe protected Space or to record video data for further detection. But watching surveillance video is a labour-intensive need to be controlled. It is also a very tedious and time-consuming job and human observers can easily lose attention.

**Disadvantage**

1. Time Consuming process.

2. More Effort.

**3.2 PROPOSED SYSTEM**

In the proposed work, Motion detection is performed by using OpenCV and Pandas library. Captured videos are treated as a stack of pictures called frames. Different frames are compared to the static frame which has no movements. We compared two images by comparing the intensity value of each pixel. In this project we have used STAE (Spatial Temporal Auto Encoder) deep learning model to predict abnormal behaviour and this model get trained on normal peoples walking videos frames and then test video will be input to this model which will analyse STAE pattern and then return the event and this event will be compared with test frame using Euclidean distance and if this distance crossed normal behaviour threshold then application will display alert message.

The system will provide an easy way to monitor the traffic and give the appropriate result, parking lots for security purposes also, visible security cameras will help you detect thieves from breaking into cars. and will help in the security field on various platforms like parking lots, home security, it will make it easy to monitor the various abnormal activities and suspicious events.

**Advantages**

1. More Security.

2. Easy to monitor.

**MODULES**

To implement this project we have designed following modules

**Upload Video Frames Dataset:** Using this module we can upload dataset video frames to application

**Dataset Preprocessing:** Using this module we will read each image and then extract each pixel and then normalize image pixel values between 0 and 1

**Train Spatial Temporal AutoEncoder Model:** In this module we will input process and normalize images to encoder model to generate STAE model

**Test Video Surveillance:** Using this module we will upload test image and then extract each frame from video and then apply STAE model on frame to predict event and this event will be compare with test frame using Euclidean distance and if this distance cross normal behaviour threshold then application will display alert message.

**3.3. PROCESS MODEL USED WITH JUSTIFICATION**

**SDLC (Umbrella Model):**

**Umbrella Activity**

**Umbrella Activity**

**Umbrella Activity**

1. Feasibility Study
2. TEAM FORMATION
3. Project Specification PREPARATION

Business Requirement Documentation

ANALYSIS & DESIGN

CODE

UNIT TEST

DOCUMENT CONTROL

ASSESSMENT

TRAINING

INTEGRATION & SYSTEM TESTING

DELIVERY/INSTALLATION

ACCEPTANCE TEST

Requirements Gathering

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Stages in SDLC:**

* Requirement Gathering
* Analysis
* Designing
* Coding
* Testing
* Maintenance

**Requirements Gathering** **stage:**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

* Feasibility study is all about identification of problems in a project.
* No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
* Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

**Analysis Stage:**

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

**Designing Stage:**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

  
When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artefacts will be produced. Software artefacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artefacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artefact is linked to a specific design element, and that each developed artefact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

**Integration & Test Stage:**

During the integration and test stage, the software artefacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

* **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artefacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

**Maintenance:**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

**3.4. Software Requirement Specification**

**3.4.1. Overall Description**

A Software Requirements Specification (SRS) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) is a complete description of the behaviour of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Non-functional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development lifecycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

* [Business requirements](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms what must be delivered or accomplished to provide value.
* Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
* Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:
* **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

* **Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

* **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

**3.4.2. External Interface Requirements**

**User Interface**

The user interface of this system is a user friendly python Graphical User Interface.

**Hardware Interfaces**

The interaction between the user and the console is achieved through python capabilities.

**Software Interfaces**

The required software is python.

**Operating Environment**

Windows XP.

**HARDWARE REQUIREMENTS:**

# Processor - Pentium –IV

* Speed - 1.1 Ghz
* RAM - 256 MB(min)
* Hard Disk - 20 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - SVGA

**SOFTWARE REQUIREMENTS:**

* Operating System - Windows7/8
* Programming Language - Python

**4. SYSTEM DESIGN**

**UML Diagram:**

The Unified Modelling Language allows the software engineer to express an analysis model using the modelling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

* + **User Model View**
    1. This view represents the system from the users perspective.
    2. The analysis representation describes a usage scenario from the end-users perspective.
  + **Structural Model view**
    1. In this model the data and functionality are arrived from inside the system.
    2. This model view models the static structures.
* **Behavioural Model View**

It represents the dynamic of behavioural as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

* **Implementation Model View**

In this the structural and behavioural as parts of the system are represented as they are to be built.

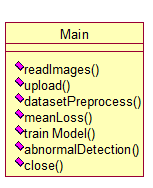
* **Environmental Model View**

In this the structural and behavioural aspects of the environment in which the system is to be implemented are represented.

**Class Diagram:**

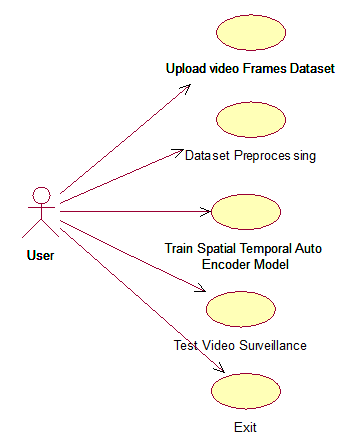
The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class
* The bottom part gives the methods or operations the class can take or undertake



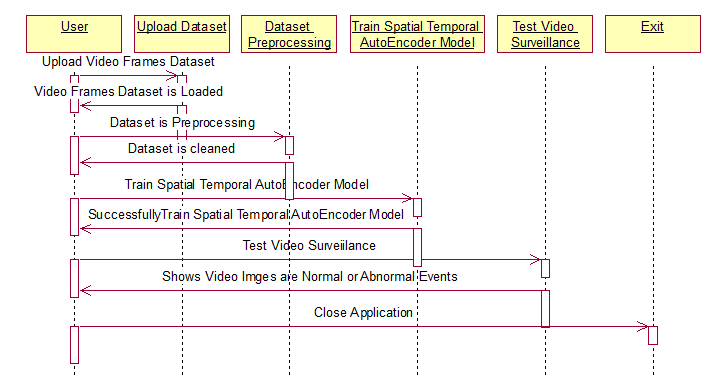
**Use case Diagram:**

A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

****

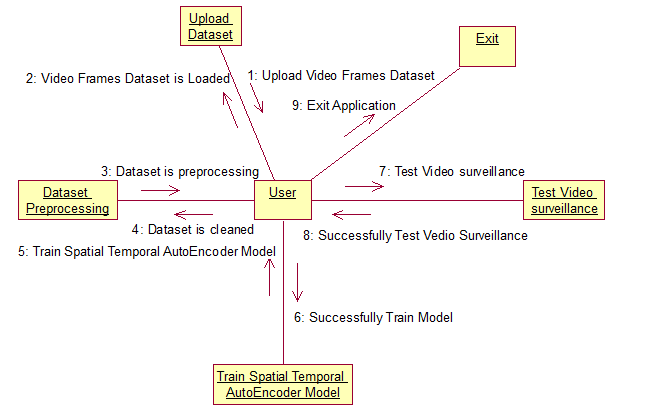
**Sequence diagram:**

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.



**Collaboration diagram:**

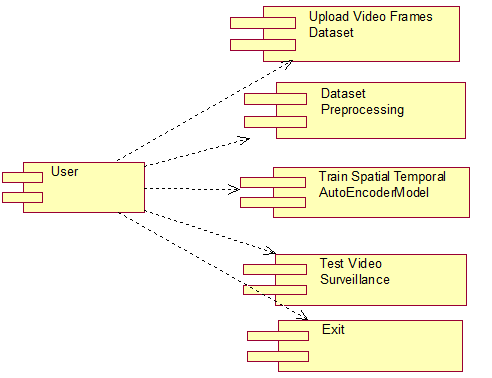
A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system.



**Component Diagram:**

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

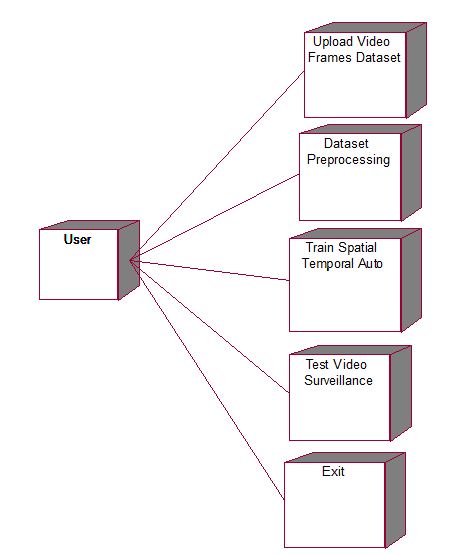
Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.

****

**Deployment Diagram:**

A **deployment diagram** in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

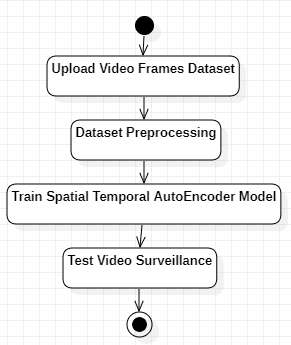
The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.



**Activity Diagram:**

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent

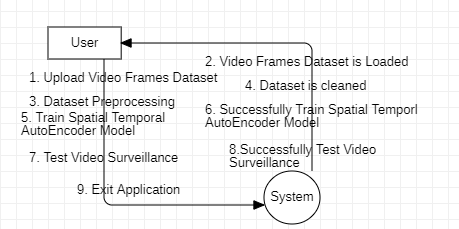
**Activity Diagram:**



**Data Flow Diagram:**

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.



**5. IMPLEMETATION**

**5.1 Python**

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

**History of Python:**

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

**Why Python was created?**

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

**Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

**Features of Python:**

**A simple language which is easier to learn**

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

**Free and open-source**

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software’s written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

**Portability**

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

**Extensible and Embeddable**

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

**A high-level, interpreted language**

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

**Large standard libraries to solve common tasks**

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQLdb library using import MySQLdb .

Standard libraries in Python are well tested and used by hundreds of people. So you can be sure that it won't break your application.

**Object-oriented**

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

**Applications of Python:**

**1. Simple Elegant Syntax**

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

a = 2

b = 3

sum = a + b

print(sum)

**2. Not overly strict**

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

**3. Expressiveness of the language**

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

**4. Great Community and Support**

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

**5.2 Sample Code:**

**Main.py**

from tkinter import \*

import tkinter

from tkinter import ttk

from keras.preprocessing.image import img\_to\_array,load\_img

import numpy as np

import glob

import os

import cv2

from keras.layers import Conv3D,ConvLSTM2D,Conv3DTranspose

from keras.models import Sequential

from keras.callbacks import ModelCheckpoint, EarlyStopping

import imutils

from keras.models import load\_model

from PIL import Image

from tkinter import filedialog

from tkinter.filedialog import askopenfilename

main = tkinter.Tk()

main.title("Intelligent Video Surveillance Using Deep Learning")

main.geometry("1300x1200")

global filename

global model

images = []

def readImages(path):

img = load\_img(path)

img = img\_to\_array(img)

img = cv2.resize(img, (227,227), interpolation = cv2.INTER\_AREA)

gray = 0.2989 \* img[:,:,0] + 0.5870 \* img[:,:,1] + 0.1140 \* img[:,:,2]

images.append(gray)

def upload():

global filename

filename = filedialog.askdirectory(initialdir=".")

text.delete('1.0', END)

text.insert(END,filename+" loaded\n");

def datasetPreprocess():

global filename, images

images.clear()

text.delete('1.0', END)

img\_list = os.listdir(filename)

for img in img\_list:

print("Dataset/"+img)

readImages("Dataset/"+img)

images = np.array(images)

testImage = images[0]

height,width,color = images.shape

images.resize(width,color,height)

images = (images - images.mean()) / (images.std())

images = np.clip(images, 0, 1)

text.insert(END,"Total images found in dataset: "+str(images.shape[0]))

cv2.imshow("Process Images",testImage/255)

cv2.waitKey(0)

def meanLoss(image1, image2):

difference = image1 - image2

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

def trainModel():

global model

text.delete('1.0', END)

if os.path.exists('model/survey\_model.h5'):

model = load\_model("model/survey\_model.h5")

else:

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'])

frames = images.shape[2]

frames = frames-frames%10

training\_data = images[:,:,:frames]

training\_data = training\_data.reshape(-1,227,227,10)

training\_data = np.expand\_dims(training\_data,axis=4)

target\_data = training\_data.copy()

callback\_save = ModelCheckpoint("model/survey\_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 = 1, epochs=5, callbacks = [callback\_save,callback\_early\_stopping])

stae\_model.save("model/survey\_model.h5")

text.insert(END,"Auto Encoder Stae model generated & saved inside model folder")

def abnormalDetection():

global model

text.delete('1.0', END)

filename = filedialog.askopenfilename(initialdir="testVideos")

cap = cv2.VideoCapture(filename)

print(cap.isOpened())

while cap.isOpened():

imagedump=[]

ret,frame=cap.read()

for i in range(10):

ret,frame=cap.read()

if frame is not None:

image = imutils.resize(frame,width=700,height=600)

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)

output=model.predict(imagedump)

loss=meanLoss(imagedump,output)

if frame is not None:

if frame.any()==None:

print("none")

else:

break

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

break

print(str(frame)+" "+str(loss))

if loss>0.00068:

print('Abnormal Event Detected')

cv2.putText(image,"Abnormal Event",(100,80),cv2.FONT\_HERSHEY\_SIMPLEX,2,(0,0,255),4)

else:

cv2.putText(image,"Normal Event",(100,80),cv2.FONT\_HERSHEY\_SIMPLEX,2,(0,255,255),4)

cv2.imshow("video",image)

cap.release()

cv2.destroyAllWindows()

def close():

global main

main.destroy()

font = ('times', 16, 'bold')

title = Label(main, text='Intelligent Video Surveillance Using Deep Learning')

title.config(bg='LightGoldenrod1', fg='medium orchid')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 12, 'bold')

text=Text(main,height=30,width=100)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=400,y=100)

text.config(font=font1)

font1 = ('times', 12, 'bold')

uploadButton = Button(main, text="Upload Video Frames Dataset", command=upload)

uploadButton.place(x=50,y=100)

uploadButton.config(font=font1)

processButton = Button(main, text="Dataset Preprocessing", command=datasetPreprocess)

processButton.place(x=50,y=150)

processButton.config(font=font1)

trainButton = Button(main, text="Train Spatial Temporal AutoEncoder Model", command=trainModel)

trainButton.place(x=50,y=200)

trainButton.config(font=font1)

testButton = Button(main, text="Test Video Surveillance", command=abnormalDetection)

testButton.place(x=50,y=250)

testButton.config(font=font1)

exitButton = Button(main, text="Exit", command=close)

exitButton.place(x=50,y=300)

exitButton.config(font=font1)

main.config(bg='OliveDrab2')

main.mainloop()

**6. TESTING**

**Implementation and Testing:**

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

## Implementation

## The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

## Testing

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

### System Testing

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

**Module Testing**

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

**Integration Testing**

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

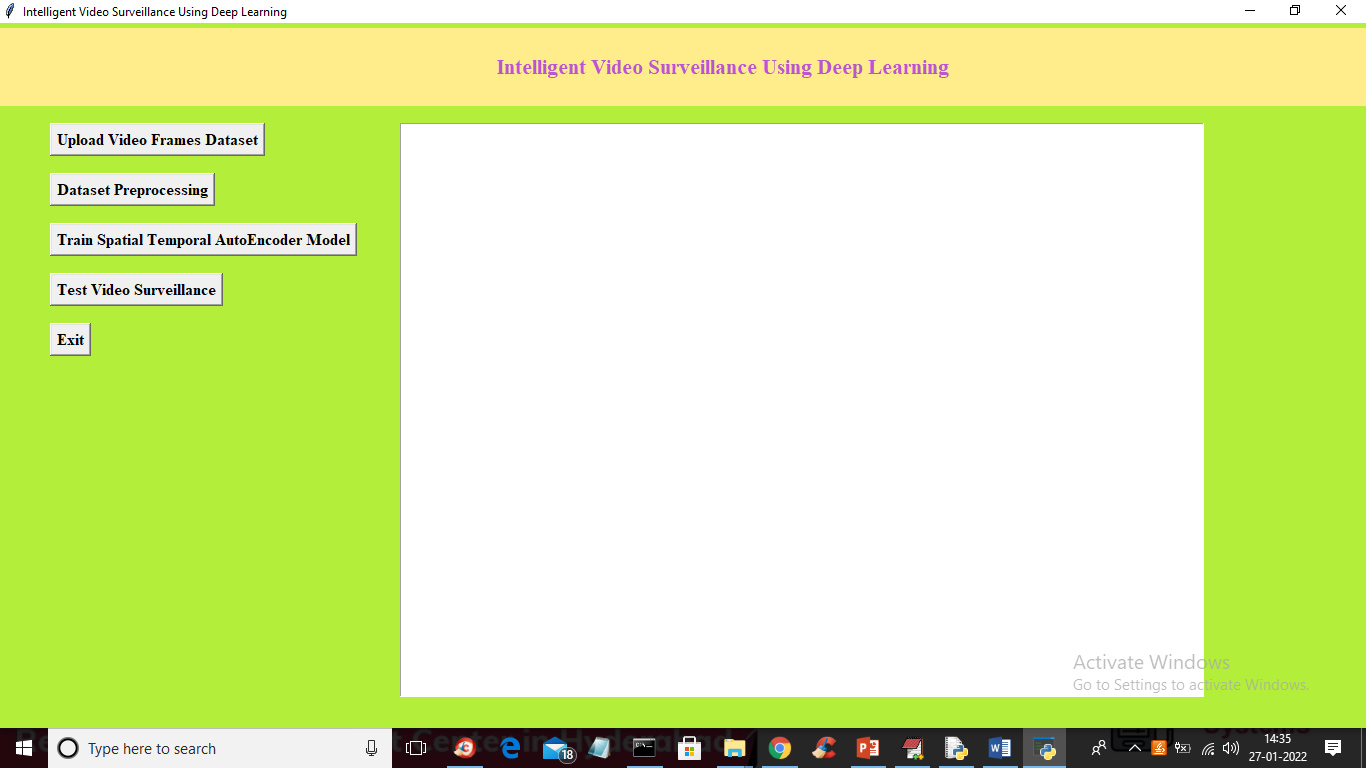
**Acceptance Testing**

When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

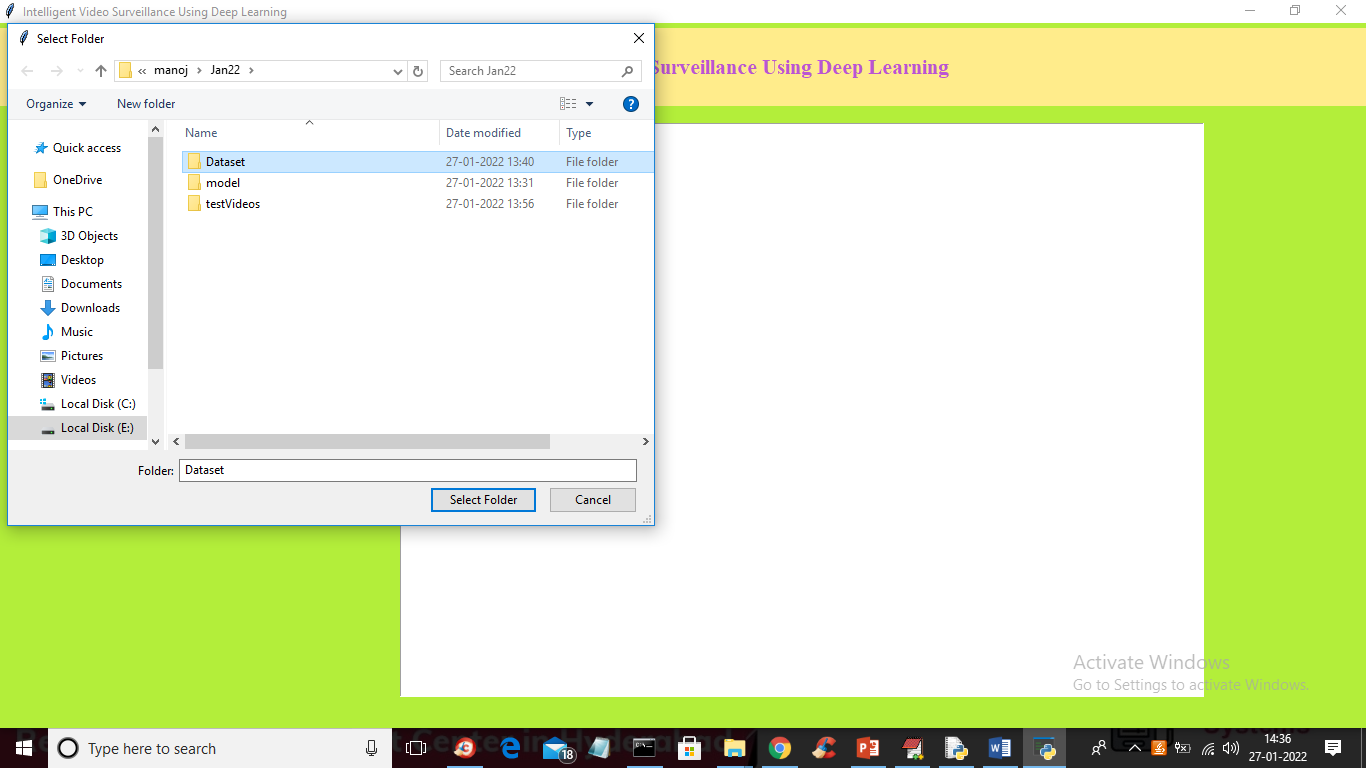
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | **Test Case Status** | **Test Priority** |
| **Step** | **Expected** | **Actual** |
| 01 | upload Video Frames Dataset | Test whether the Video Frames Dataset is uploaded or not into the system | If the Video Frames Dataset may not uploaded | we cannot do further operations | Video Frames Dataset uploaded we will do further operations | High | High |
| 02 | Pre-processing Dataset | Test whether the Video Frames Dataset is uploaded or not into the system | If the Video Frames Dataset may not uploaded | we cannot do Pre-processing the Dataset | Video Frames Dataset uploaded we will do pre-processing Dataset | High | High |
| 03 | Train Spatial Temporal Auto Encoder Model | Verify the Spatial Temporal Auto Encoder Model will Trained or not | Without training model | we cannot run Spatial Temporal Auto Encoder model | we can run Spatial Auto Encoder Model | High | High |
| 04 | Test Video Surveillance | Verify the Dataset is Uploaded or not | Without Uploading Images | we cannot Test Video Surveillance | we can Test Video Surveillance | High | High |

**7. SCREENSHOTS**

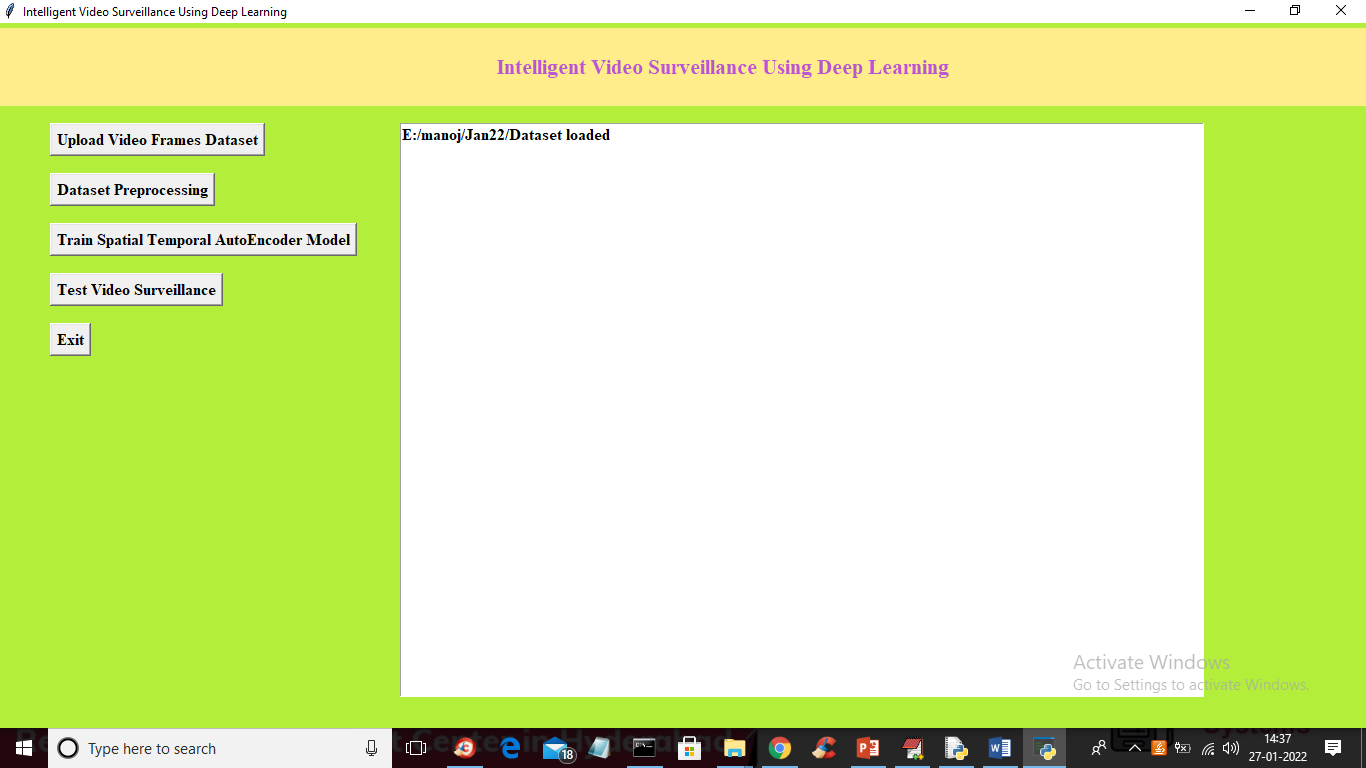
To run project double click on ‘run.bat’ file to get below screen



In above screen click on ‘Upload Video Frames Dataset’ button to upload dataset and to get below screen



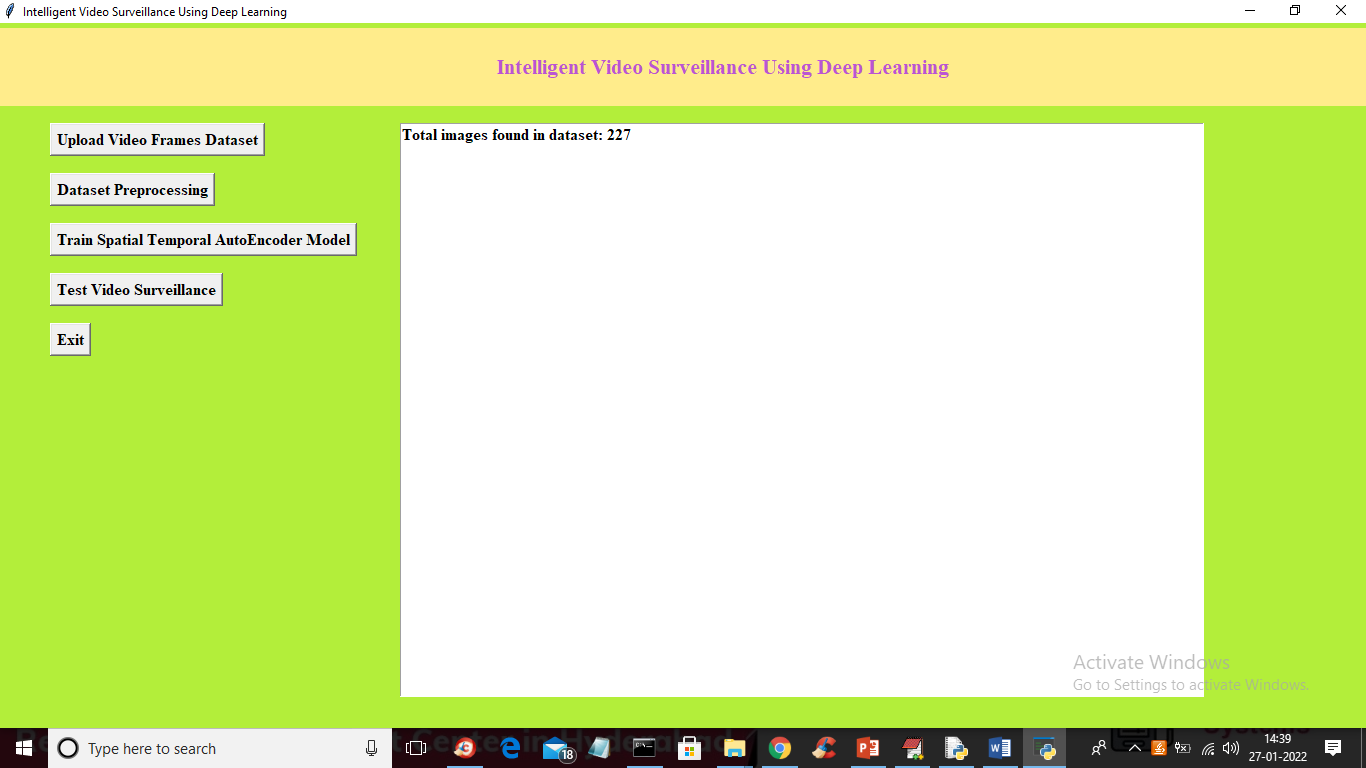
In above screen selecting and uploading ‘Dataset’ folder and then click on ‘Select Folder’ button to load dataset and to get below screen



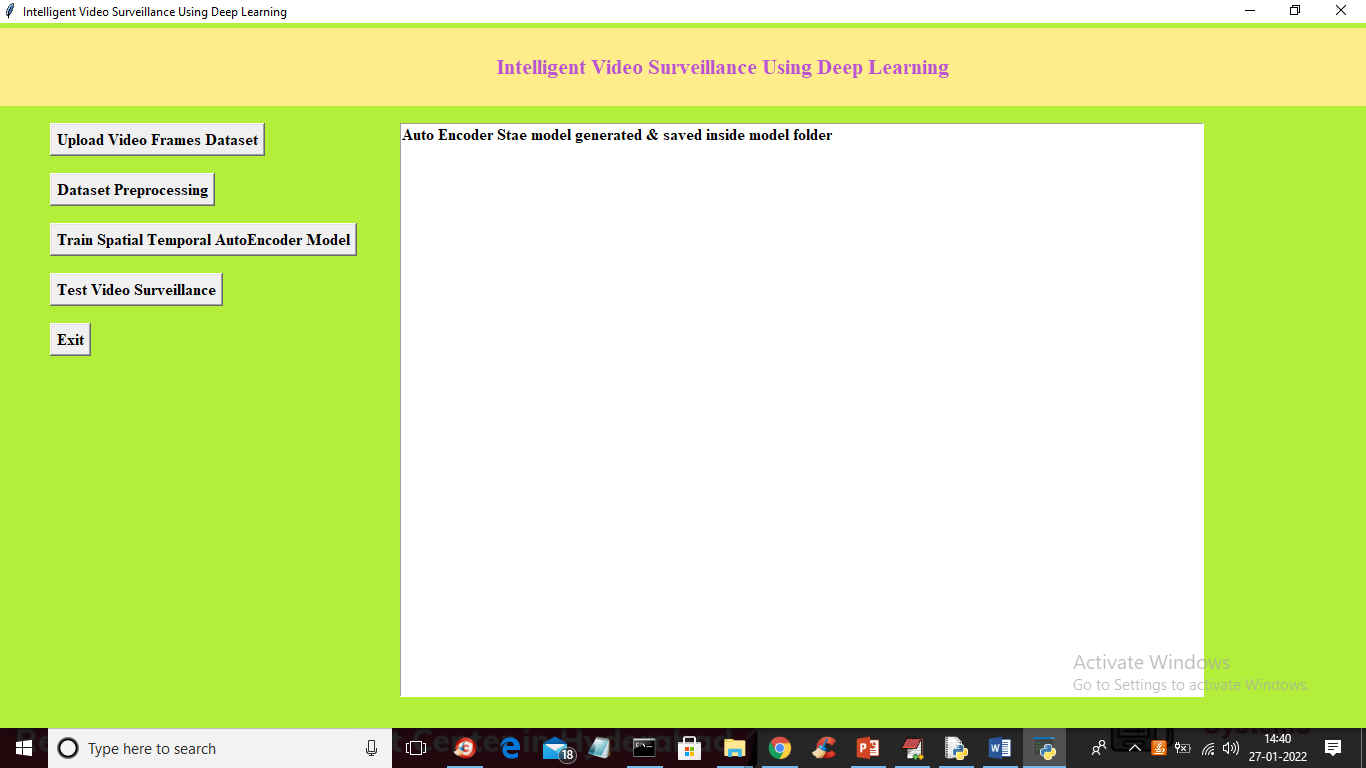
In above screen dataset loaded and now click on ‘Dataset Preprocessing’ button to normalize video frames and to get below screen



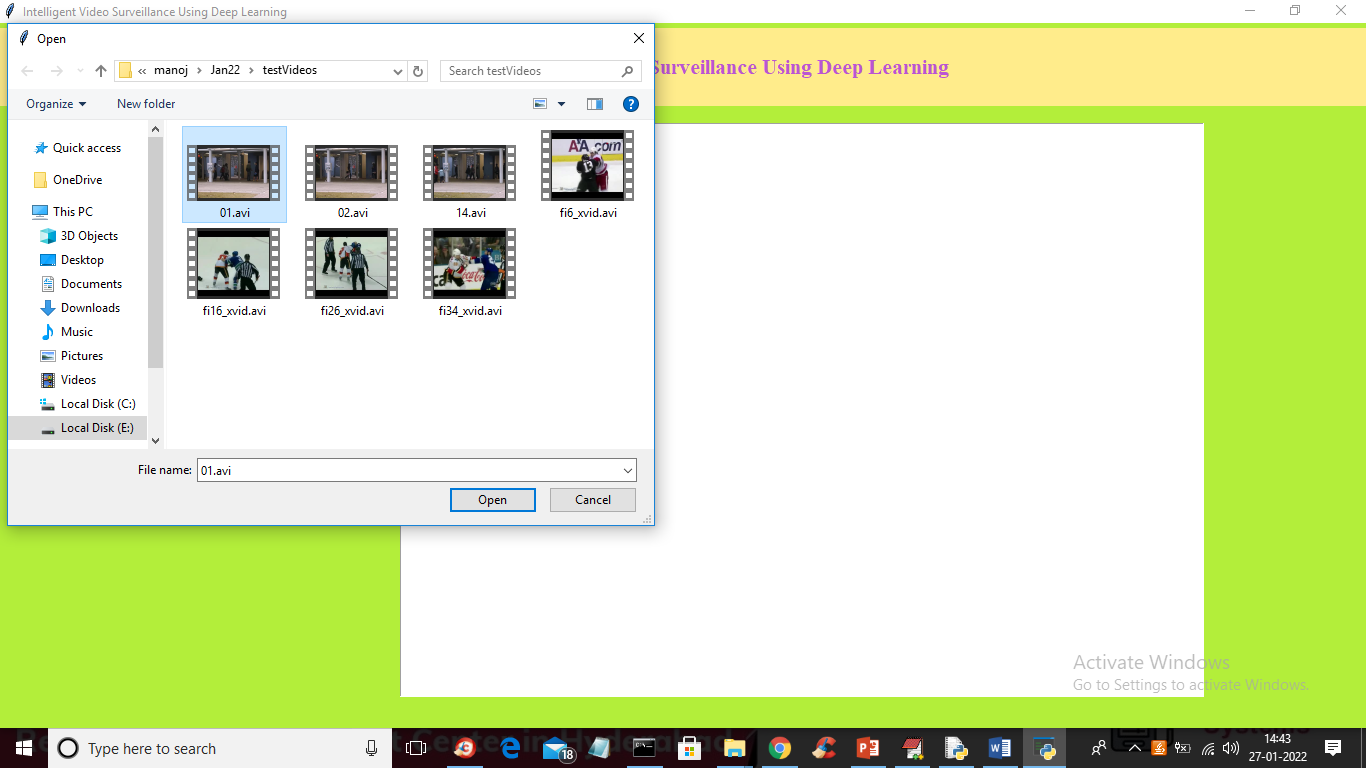
In above screen all images are processed and I am displaying one sample image to see all images are process normally and now closed above image to get below output



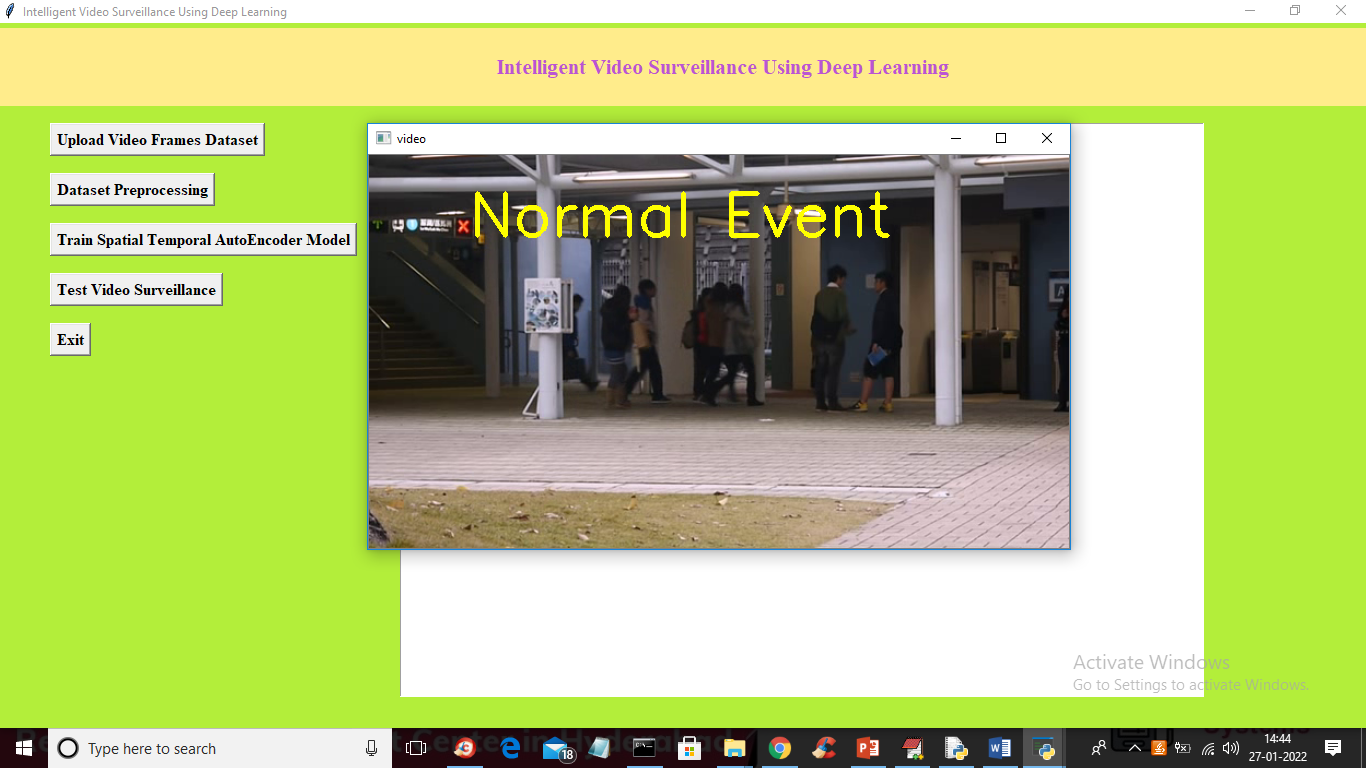
In above screen we can see dataset contains 227 image and all images are processed and now click on ‘Train Spatial Temporal AutoEncoder Model’ button to train STAE model with process images and to get below output

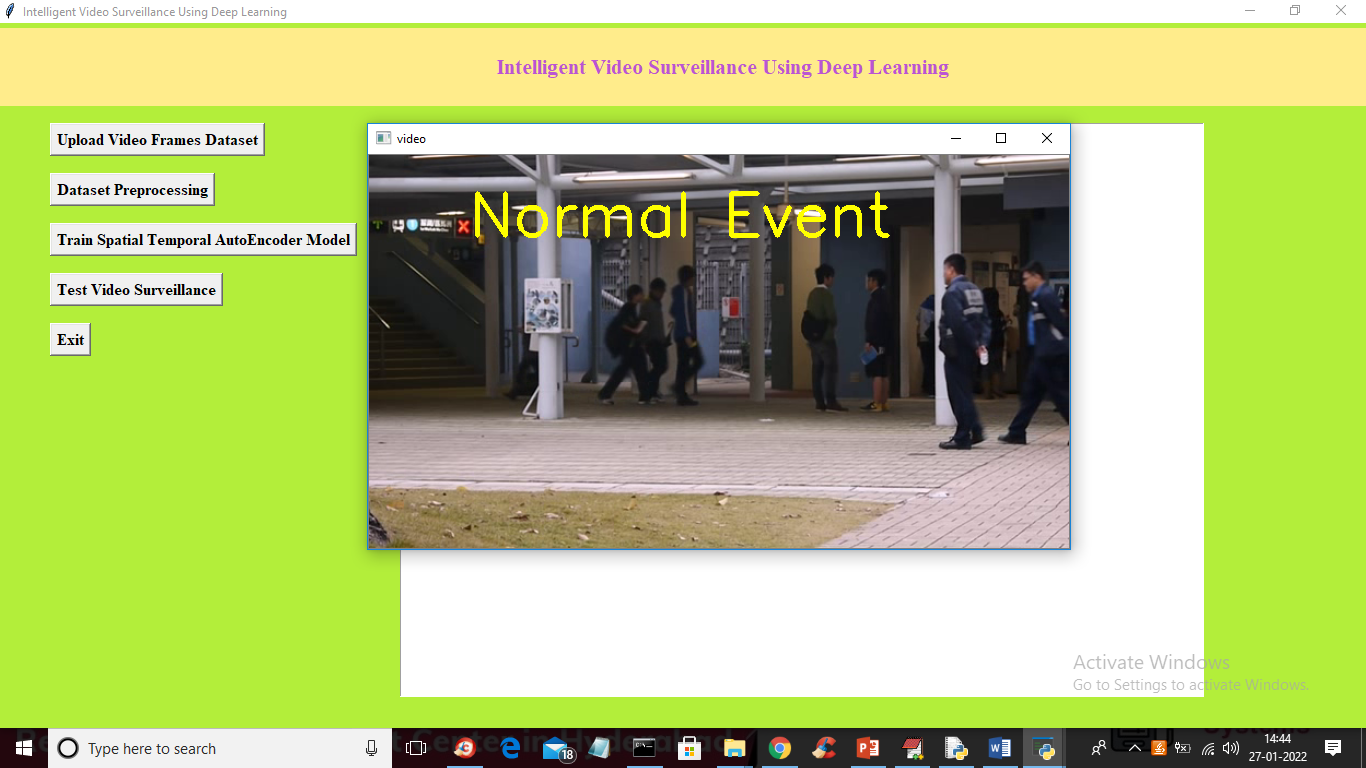


In above screen STAE model generated and now click on ‘Test Video Surveillance’ button to upload test video and to get below output

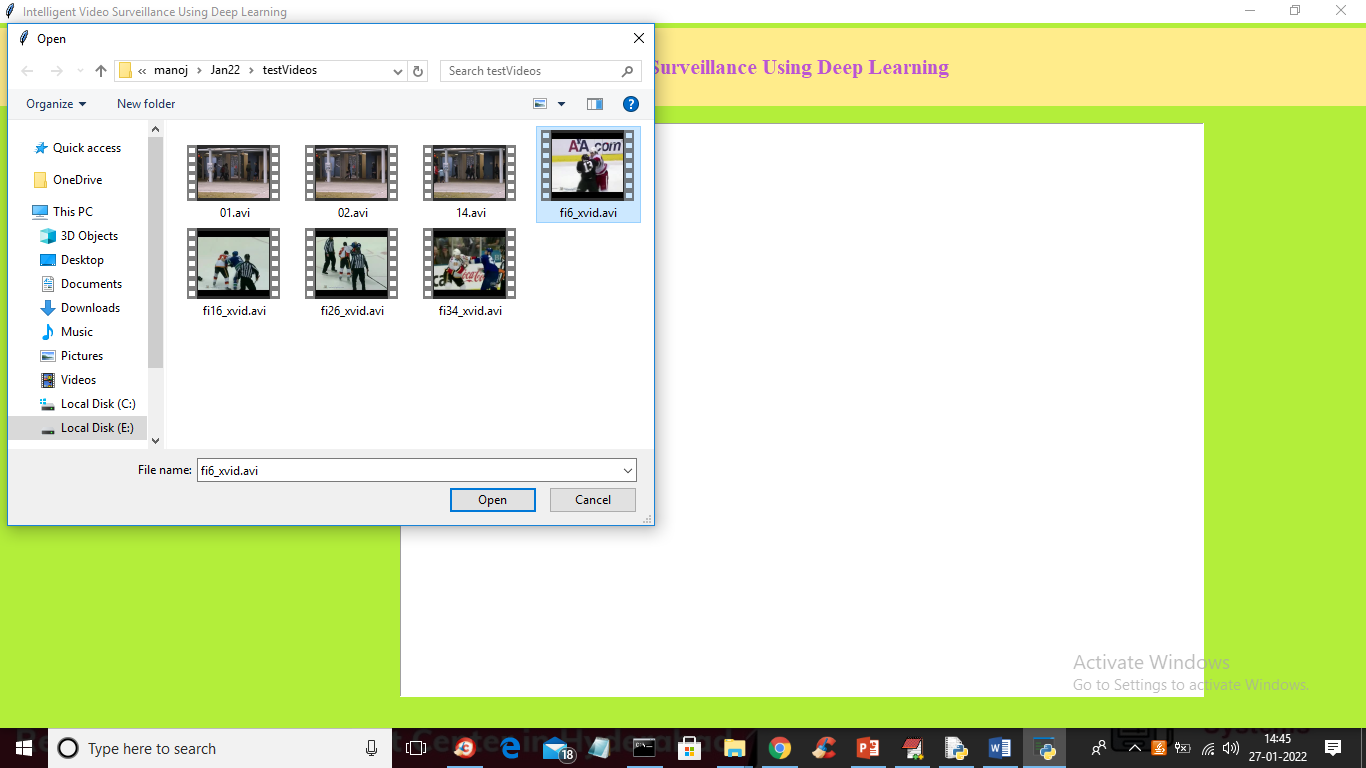


In above screen selecting and uploading ’01.avi’ video file and then click on ‘Open’ button to upload video and to get below output

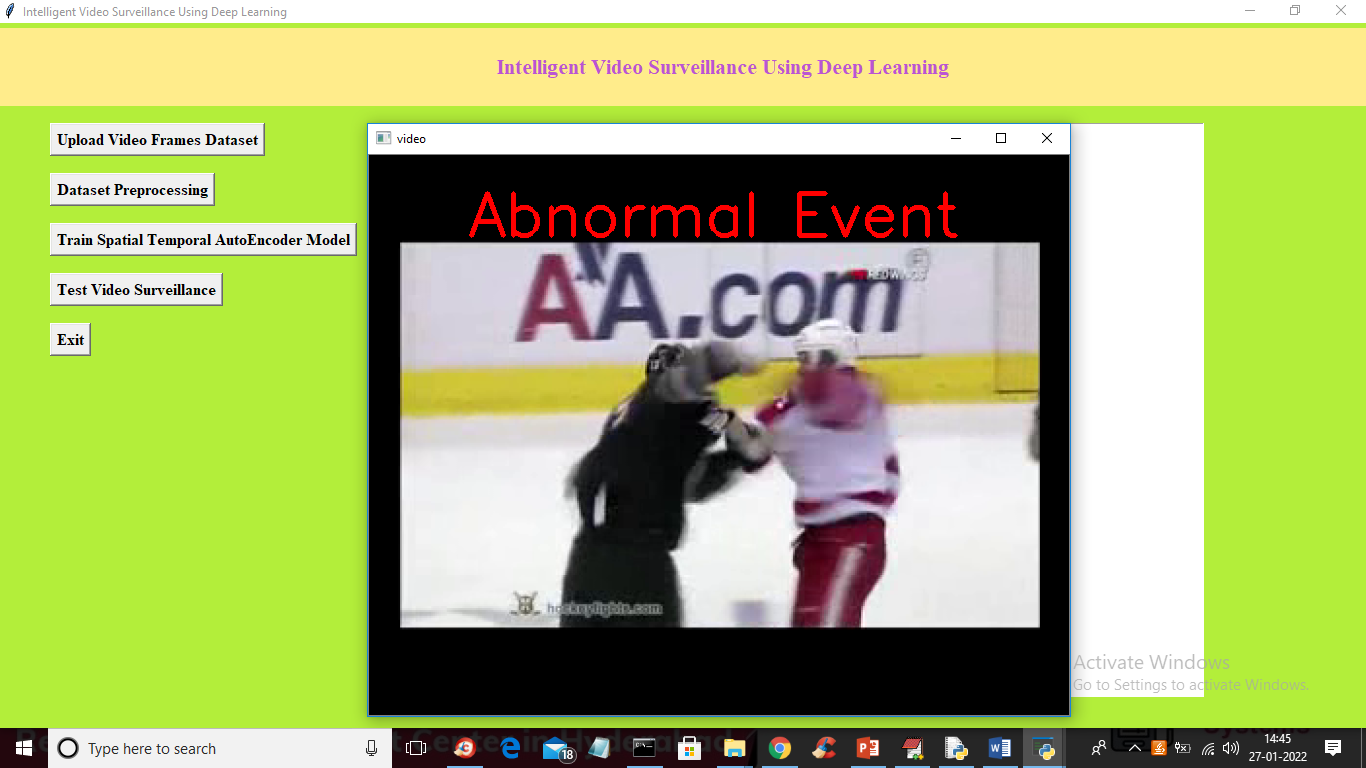




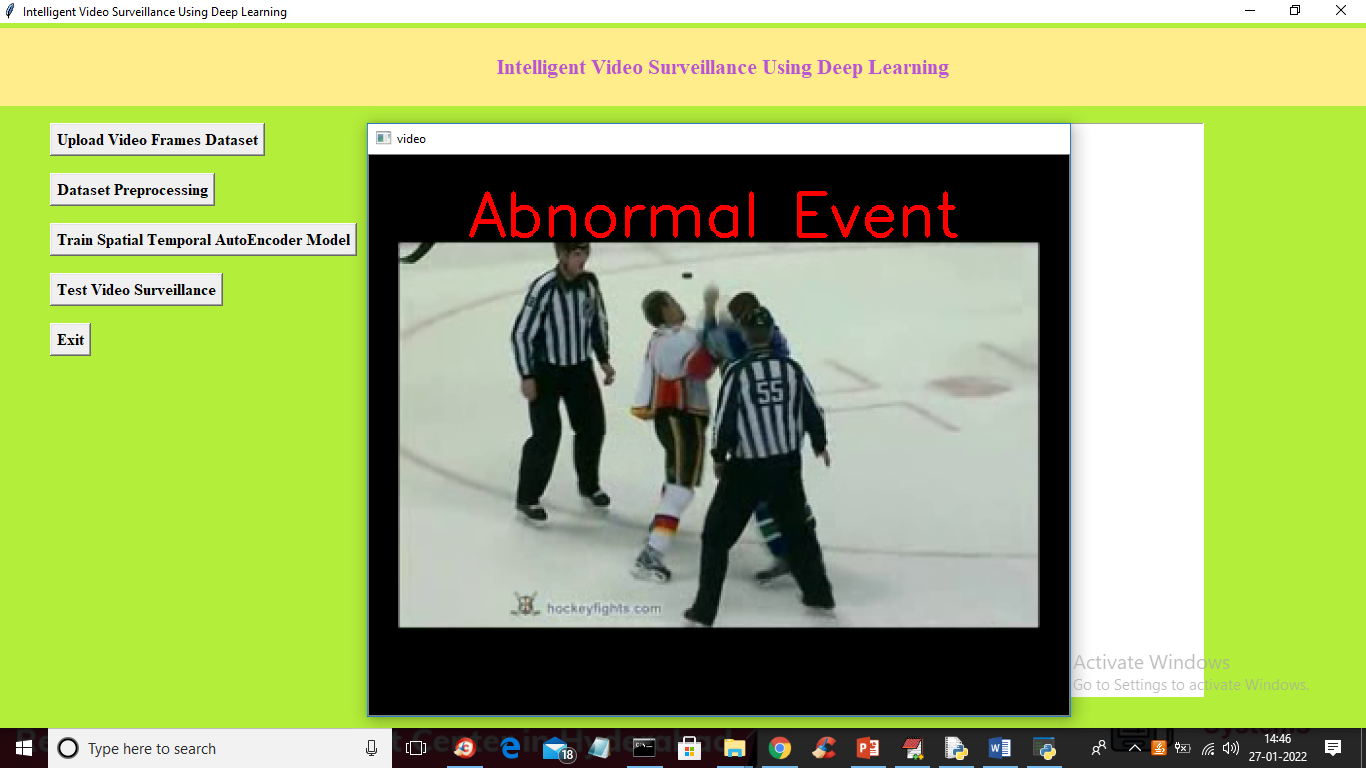
In above screen peoples are just walking so its consider as Normal Event and now press ‘q’ key to close video and upload another video



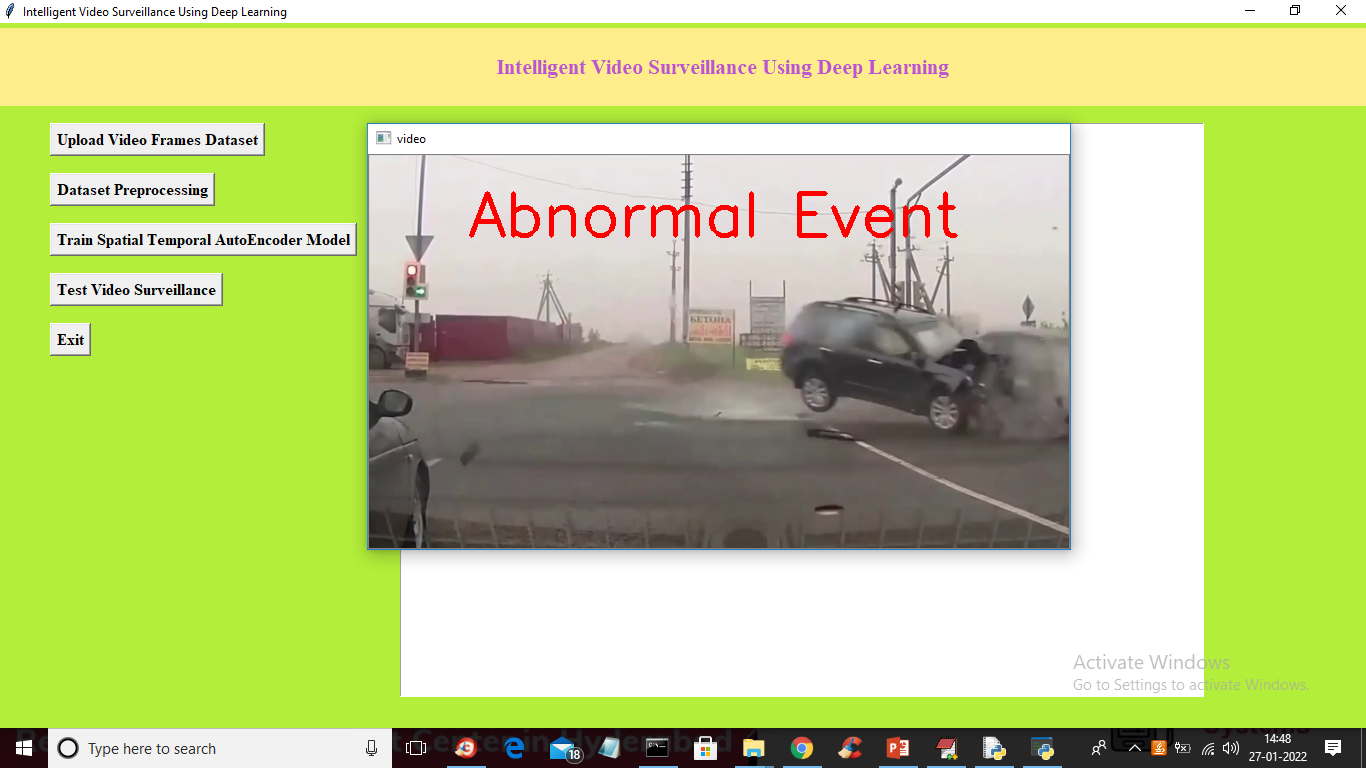
In above screen uploading another video and then click ‘Open’ button to get below output



In above screen two peoples are fighting so its not normal walk so displaying alert as abnormal event and below is other output



Similarly u can upload any video and test it



**8. CONCLUSION**

The paper reviews intelligent surveillance video analysis techniques. Reviewed papers cover wide variety of applications. The techniques, tools and dataset identiﬁed were listed in form of tables. Survey begins with video surveillance analysis in general perspective, and then ﬁnally moves towards crowd analysis. Crowd analysis is diﬃcult in such a way that crowd size is large and dynamic in real world scenarios. Identifying each entity and their behaviour is a difficult task. Methods analyzing crowd behaviour were discussed. The issues identiﬁed in existing methods were listed as future directions to provide eﬃcient solution.

**9.REFERENCES**

[1] VRIGKAS, Michalis, NIKOU, Christopher’s, ET KAKADIARIS, Iohannis A. A review of human activity recognition methods. Frontiers in Robotics and AI, 2015, vol. 2, p. 28.

[2] POPULAR, Oluwatoyin P. ET WANG, Kejun. Video-based abnormal human behavior recognition—A review. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2012, vol. 42, no 6, p. 865-878.

[3] BENABBAS, Yassin, IHADDADENE, Nacim, ET DJERABA, Chaabane. Motion pattern extraction and event detection for automatic visual surveillance. EURASIP Journal on Image and Video Processing, 2010, vol. 2011, no 1, p. 163682.

[4] HUNG, Ya-Xuan, CHIANG, Chih-Yen, HSU, and Steen J., et al. Abnormality detection for improving elder's daily life independence. In: International Conference on Smart Homes and Health Telematics. Springer Berlin Heidelberg, 2010. p. 186-194.

[5]ADAM, Amit, RIVLIN, Ehud, SHIMSHONI, Ilan, et al. Robust real-time unusual event detection using multiple fixed-location monitors.

[6] SHIN, Jae Hyuk, LEE, Boreom, ET PARK, Kwang Suk. Detection of abnormal living patterns for the elderly living alone using support vector data description. IEEE Transactions on Information Technology in Biomedicine, 2011, vol. 15, no 3, p. 438-448.

[7] PALANIAPPAN, Adithyan, BHARGAVI, R., ET VAIDEHI, V. Abnormal human activity recognition using SVM based approach. In: Recent Trends in Information Technology (ICRTIT), 2012 International Conference on. IEEE, 2012. p. 97-102.

[8] A. Datta, M. Shah and N. Lobo, “Person-on-person violence detection in video data”, Proceedings of the 16th International Conference on Pattern Recognition, vol. 1, pp. 433-438, 2002.

[9] Paolo Rota, Nicola Conci, Nicu Sebe, and James M Rehg, "Real-life violent social interaction detection", International Conference on Image Processing, pp. 3456– 3460, 2002.

[10] M. Drulea, and S. Nedevschi, “Total Variation Regularization of Local-Global Optical Flow”, IEEE Int. Con. Intelligent Transportation Systems, pp. 318-323, 2011.

[11] E. ESEN, M. ARABACI, and M. SOYAL,”Fight Detection in Surveillance Videos”, Int. Workshop on Content-Based Multimedia Indexing, pp. 131-135, Jun 2013.

[12] CAVIAR Test Case Scenarios, http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1 Daniel Gutchess, M Trajkovics, Eric Cohen-Solal, Damian Lyons, and Anil K Jain, “A background model initialization algorithm for video surveillance,” in ComputerVision, 2001. ICCV 2001. Proceedings. EighthIEEE International Conference on. IEEE, 2001, vol. 1, pp. 733–740.

[13] Varun Chandola, Arindam Banerjee, and Vipin Kumar, "Anomaly detection: A survey," ACM computing surveys (CSUR), vol. 41, no. 3, pp. 15, 2009.

[14] Yael Pritch, Alex Rav-Acha, and Shmuel Peleg, "Nonchronological video synopsis and indexing," IEEE transactions on pattern analysis and machine intelligence,vol. 30, no. 11, pp. 1971–1984, 2008.

[15] Chris Stauffer and W Eric L Grimson, “Adaptive background mixture models for real-time tracking,” in Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on. IEEE, 1999, vol. 2, pp.246–252.

[16]Olivier Barnich and Marc Van Droogenbroeck, "Vibe: A universal background subtraction algorithm for video sequences," IEEE Transactions on Image processing, vol. 20, no. 6, pp. 1709–1724, 2011.

[17] Pierre-Luc St-Charles, Guillaume-Alexandre Bilodeau, and Robert Bergevin, “Subsense: A universal change detection method with local adaptive sensitivity,” IEEE Transactions on Image Processing, vol. 24, no. 1, pp. 359– 373, 2015.