REAL-TIME DATA MINING FOR MULTIMEDIA STREAMING

by

Sepideh Banihashemi

Master of Information Technology Management

Shahid Beheshti University, 2015

A thesis

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Master of Science

in the program of

Computer Science

Toronto, Ontario, Canada, 2018

Author’s Declaration

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

REAL-TIME DATA MINING FOR MULTIMEDIA STREAMING

Master of Science, 2018

Sepideh Banihashemi

Computer Science

Ryerson University

Abstract

Developing a Web Video Player connected to a security surveillance camera for collecting the video streams is the main objective of this study. The Developed Web Application tracks the target object through the sequences of video frames and generates the object trajectories. The video frames are analyzed, and the object trajectories are fed into a classifier or clustering method for training and movement detection purposes. In this thesis, several machine learning techniques are applied and implemented in Batch and in Real-Time mode including SVM, J48 Decision Tree, PART, Decision Table, Decision Stump, Multilayer Perceptron, and K-Means clustering by using two customized datasets. The object tracking, and movement detection are based on a simplified HSV color space model. The developed Web Application and proposed architecture are implemented on a local area network with in-house Server as well as a single computer and can detect the trajectories of the moving objects effectively.

Acknowledgements

First and foremost, I would like to express my special sincere thanks to my supervisor, Professor Abdolreza Abhari, for all the opportunities I was given by him and assigning this project to me which is intended to improve the security systems in residential and commercial buildings, airports and all places in which the security cameras are installed.

I had the opportunity to work as part of a great team in our research lab during my thesis. I would like to thank all my colleagues in DSMP lab, especially Jason Li, and Steve Kanellis for providing valuable information and technical help.

Nobody has been more important to me in the pursuit of this project than the members of my family. I would like to thank my parents whose love, support, and guidance are with me in whatever I pursue.

Table of Contents

List of Tables

List of Figures

List of Acronyms

DOM	Document Object Model
DTL	Decision Table Local
DTM	Decision Table Majority
FN	False Negative
FP	False Positive
GUI	Graphical User Interface
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HSV	Hue, Saturation, Value
J2EE	Java 2 Enterprise Edition
JRE	Java Runtime Environment
JVM	Java Virtual Machine
KMS	Kurento Media Server
MLP	Multi-Layer Perceptron
PC	Personal Computer
PHP	Hypertext Preprocessor
QP	Quadratic Problem
RBF	Radial Basis Function
RGB	Red, Green, Blue
RI	Rand Index
ROC	Relative Operating Characteristic
ROI	Region of Interest
RTC	Real-Time Multimedia Communication
RTSP	Real-Time Streaming Protocol
SVM	Support Vector Machines
TP	True Positives

Chapter 1: Introduction

1.1. Motivation

Data mining for Multimedia data is a new and an ongoing research area which is about detecting the patterns in Multimedia data. Multimedia data includes image, audio and video data. Transforming the video data into a digital representation can be performed by creating a sequence of video frames. The term data mining itself denotes knowledge discovery which has focused more on multimedia data due to its existence everywhere and evolution of structured to unstructured data, such as image, audio, and video data [1]. Image processing is different from data mining or pattern recognition of images (referred to image data mining in this thesis). The difference is, image processing deals with finding and extracting the particular features in an image, whereas image data mining is about finding useful patterns in a set of images or video frames. In addition, the goal in pattern recognition is to recognize the significant patterns but the objective of image mining is to create the considerable patterns without having any previous information about the existing models in the database of images. Image data mining is an interdisciplinary research area in computer vision, image processing, machine learning, image retrieval, artificial intelligence, and data mining [2]. One of the applications of a security surveillance camera is to detect the abnormal activities and behaviors in which the user is notified when an anomaly occurs [3]. A Real-Time automatic tracking system needs to be implemented for observing the object behaviors since a human operator is not available either to monitor a large amount of video streams in Real-Time in order to detect the anomalous event or to look for the proper video image after the incident has happened [4].

This thesis proposes image data mining in which we plan to apply data mining methods for tracking the object of interest through the video frames collected by a security surveillance camera and streamed by a web media server. The video frames will be analyzed to collect the object of interest’s movements and then fed into another model for training and detecting the movement. The main motivation of this work is to propose an architecture for performing Real-Time data mining for multimedia data.

1.2. Problem Statement

In this thesis, we have investigated the following research questions:

1. What architecture/web application can be used to collect the trajectories of a moving object by using a security surveillance camera installed on an in-house network in Real-Time?
2. What machine learning approaches can provide a more accurate and faster way for detecting the trajectory of a moving object in Real-Time mode and Batch mode and how they can be used in anomaly detection?
3. What architecture/web application is more effective for machine learning approaches when trained in Real-Time with provided labeled and unlabeled datasets?
4. What architecture/web application can support the anomalous event detection automatically by eliminating the human interaction and sending notifications to the user in case of anomaly occurrence?

1.3. Objectives and Challenges

The initial objective of this thesis is to develop a Web Application for a Web Video Player presented on a client's machine which communicates with a Video Streaming in-house Server that security surveillance cameras connect to. It is clear that we don't want to use the available commercial products such as NVR [5] as we want that the camera connects directly to the server not to any other devices such as NVR devices. In the first phase, we were searching for a network architecture in which the recorded multimedia by camera via RTSP protocol is accessible by a Web Video Player where a user can identify the object of interest. In this architecture a server processes the data and records the trajectories or sends notifications to client's machine (for example sending an alarm in case of anomaly occurrence). The available commercial products that we have reviewed, either don't have such capabilities or if they have part of these features, they use specific hardware or software such as NVR or DVR with an embedded operating system [5]. In the second phase, a model is developed by collected data to identify the trajectory patterns by clustering and classification data mining methods. In this thesis, K-means clustering and several classification methods such as SVM, PART, J48, Decision Stump, Multilayer Perceptron, and Decision Table are applied for training and test purposes. The K-means clustering is developed to detect abnormal trajectories for unlabeled movements by finding the outliers in each cluster. By finding the abnormal trajectories, one can detect the unusual and suspicious behaviors. In this work, we implemented the K-means clustering in Real-Time to prove the feasibility of this approach but left the anomaly detection for the future works. In the area of Supervised Machine Learning which employs classification techniques, first the classifier is trained by one of the labeled training datasets and then the accuracy of the classifier is measured for evaluating the performance. Two different datasets are generated for this purpose and each of them has 150 instances for training and 30 instances for the test purposes. This research proposes a framework to detect the object movement in Real-Time on the local area networks similar to the ones installed in residential/businesses buildings and high rises.

Another application of this research is to enhance the developed software and using data mining for Real-Time image processing of numerous video data to find more patterns for detecting and tracking a moving object in order to identify the suspicious behaviors for security purposes. When a domain expert is unavailable, clustering can be used by implementing K-Means clustering in the Developed Web Application.

1.4. Methodology and Implementations

What is being proposed in this research is the development of a Web Video Player on a desktop computer communicated with a Video Streaming in-house Server to which a security surveillance camera connects for capturing the video frames. The object trajectories that are generated by moving the object of interest are used in a Web Application to perform Real-Time Machine Learning detection by SVM classification and K-means clustering during multimedia streaming.

In addition, for Real-Time videos, the object tracking and detection are achieved based on the color of the target object. The object trajectories which are collected by Real-Time object movement can be used for Real-Time K-means clustering and SVM classification as well as Batch or offline mode. In SVM Real-Time, the classification is performed for only an individual trajectory at a time, then the trajectories and the detected movement types (labels) are stored in a separate file to feed into SVM Batch. After making several Real-Time videos by performing SVM batch, the accuracy of the classifier is measured both in offline and online video streaming.

Before creating any Real-Time movement, the classifier needs to be trained by one of the datasets created in our lab. In total, two data sets have been created: one object movements with known source, destination and orientation and another one, random movements in different orientations to measure the accuracy of clustering and classification methods.

To implement this application, a Web Video Player is developed on the client's machine that communicates with the video streaming server. The video streaming server connects to the security surveillance cameras for receiving data from them. The client's machine can be a simple desktop computer, which shows what the security surveillance camera captures. The server is responsible for processing data to provide simple functionalities as follows:

- The coordinates are sent to the server by client's machine and software gets input based on the coordinates of the object of interest obtained via drawing a polygon by the user; and,
- The software will continuously keep track of the object and notify the user if the object has moved out of the frame.

By drawing a polygon around the object of interest, the Web Application starts to track the target object and generates the object trajectories. These object trajectories along with the name of the movement are saved in the separate files both for training and test purposes. All the object trajectories have object coordinates formed on the movement and direction. Object Tracking and detection that is used, is a simplified color-based detection, meaning that by giving the values for a particular color, the tracker is able to track the object. Simple image processing technique was used in this work because the focus was to create a web-based architecture in which all the processing to be done in the server and a client can use a system with a simple browser. Since a camera is connected to a server because of security purposes instead of using outside cloud computing facilities, all developments are done for an in-house Server which is a favorite architecture in the buildings using the security cameras.

1.5. Contributions

The main contributions of this thesis are listed as follows:

- Developed an in-house network architecture to detect and track the object of interest via a web browser and generate the object trajectories in the server side to be classified in Real-Time and detect the movement type that is being captured by using an IP Camera.

- A Web Application was developed to detect and track the object of interest through a web browser by using RTSP/WebRTC protocols to address research question one. This Web Application can be used for Batch and Real-Time data mining for trajectory detection.
- Two data sets were created from hundreds of videos, including three types of movements with known and unknown source and destination. These video files and data sets which contain labels can be used by other researchers for research purposes and are accessible to the public in [6].
- Several Machine Learning approaches for detecting the moving object trajectories in Batch and Real-Time modes are compared. Acquired by observation, the conclusion was to detect the object trajectories of a moving object, one can use PART, J48 Decision Tree, Decision Stump, Decision Table, and Multilayer Perceptron for Batch mode and the K-means clustering and SVM for Real-Time mode in the Web Applications that employed the security surveillance cameras.

1.6. Thesis Organization

The remainder of this thesis is organized as follows: Chapter 2 provides background information and related works which have been performed in the field of object detection and tracking. There is also an introductory information including Real-Time Streaming Protocol, movement detection based on object’s color, and OpenCV. Chapter 3 presents a detailed description of the system requirements of the proposed methodology for Real-Time data mining of multimedia streaming. Chapter 4 discusses the results of this study as well as the implementation of data mining models in Batch mode and Real-Time mode along with comparing the evaluation metrics for machine learning methods. Finally, Chapter 5 draws conclusions regarding the research findings and provides guidelines for future studies.

Chapter 2: Literature Review

2.1. Background Information

2.1.1. Ubuntu Operating System

Ubuntu is an operating system for desktop computers, cloud and internet connected things which is an open-source software [7]. This operating system is secure and accessible that enables its usage for all the people with any gender, nationality or disability [8].

Ubuntu is based on Debian GNU/Linux distribution which means that, Linux operating system did not exist on a single CD or on several of the disks that we currently use to install. Instead, this operating system contained a great deal of separate programs where each was created by a separate person and distributed differently. It took a considerable amount of time to install each of these essential programs and sometimes it required some clues and hints for resolving installation difficulties. The team of Ubuntu provides GNU, Linux and many more applications under an individual installer [9]. The kernel of the Linux which was generated by Torvalds and the GNU application that was created by Stallman are two distinct projects, but some parts of the GNU project were taken by Linux. That is the reason why most of the people indicate Linux as GNU/Linux [10].

Ubuntu can be considered as one of the very best versions of Linux which is based on Debian distribution and is suitable for both novice and professional users. Ubuntu operating system allows anyone to use simply this version of Linux, especially for Windows or Mac OS users. It supports a lot of languages as well as accessibility tools. The meaning of Ubuntu comes from an African root which translates as “Humanity to other individuals”. Ubuntu’s archive is a collection of Ubuntu derivatives and it consists of any packages which create distributions like Ubuntu, Kubuntu, Ubuntu Server, etc. Ubuntu Server is a group of packages from this archive and its installer installs a set of default packages. There are some differences between Ubuntu Server Edition and Ubuntu Desktop Edition such as Kernel distinctions. LTS version of Ubuntu means long-term support [10].

The Installation of Ubuntu Desktop is straightforward and simple compared to Windows operating system. The following steps show the installation process of Ubuntu operating system [11] :

Step 1: Inserting the Ubuntu CD/DVD which is also called an Ubuntu live disc. Set the bootable disc for CD/DVD drive in BIOS.

Step 2: Choosing the language: There is a list of all the languages which are supported by Ubuntu.

Step 3: Selecting the option: Try Ubuntu First Without Making Changes to the Computer. After choosing this option, Ubuntu starts storing the files on the RAM memory.

Step 4: Trying different items such as Sound, Video and Network Connectivity.

Step 5: Partitioning the Hard Drive: The most usual file system for Linux is ext3. Linux also requires a swap partition that is quite a small section of the hard drive operating as extra memory.

Step 6: Installing Ubuntu files on the Hard Drive: Just by clicking the Install icon on the desktop, the process is launched.

Step 7: Logging In.

Additionally, the Ubuntu operating system can be booted and installed from a live USB drive or it can be run on a Virtual Machine such as VirtualBox or VMWare Workstation Pro. The following steps describe how to install Graphical User Interface (GUI) on Ubuntu Server 14.04 LTS. However, there are some disadvantages of installing GUI on Ubuntu Server such as, making it unsecure and slower. In other words, by installing Ubuntu Server GUI, graphical tools can be used.

In order to update and upgrade the desktop, the Software Updater should be typed in the search bar. By clicking on the Software Updater, a dialog box will be shown and Install Now needs to be selected to let the update process begin. It is worth mentioning that the need for the Ubuntu operating system 14.04 LTS (64 bits) is that Kurento Media Server should be installed on this operating system [12].

2.1.2. Kurento Media Server (Release 6.6.1)

Kurento is a WebRTC Media Server and is responsible for media transmission, processing, loading and recording. Desktop computers and Smartphone platforms are dealing with Real-Time communications and browsers develop the way users access Internet services. However, there is a problem in Real-Time multimedia communication which is the fragmentation that can be created because of the lack of interoperability. Users deal with several services for communication purposes. WebRTC is one of the solutions for overcoming fragmentation by supporting standard protocols, video codecs and formats to offer interoperable multimedia communications [13].

Kurento Media Server includes pluggable modules which can be activated and deactivated. This Media Server has the following additional features compared to the usual WebRTC Media server: augmented reality, mixing, analyzing, and adjustable processing. Ubuntu 14.04 LTS (64 bits) is necessary if Kurento Media Server is required to be installed. Kurento is one of the best technologies for multimedia communications and consists of two layers which are also called planes: Signaling plane and Media plane. Signaling plane is responsible for communication management and Media plane controls media transport, media encoding/decoding, etc. Kurento Media Server is used for media encoding and media decoding as well as processing the media [14].

Kurento Media Server is based on WebRTC and consists of a group of clients' APIs which facilitates the establishment of developed video applications for web and smartphones. The characteristics of the Kurento Media Server are: communications, transcoding, recording, mixing as well as computer vision and video indexing, and speech analysis. The main objective of WebRTC is to offer Real Time multimedia Communications (RTC) on the WWW which also supports open standards and includes open source software implementations. WebRTC has been integrated into Kurento Media Server to convert RTSP to WebRTC protocols [15].

The Kurento Media Server has four principles as follows: 1) Openness: the Kurento technologies are Open Source Software; 2) Simplicity: Kurento's major feature is simplicity; 3) Standards: the main parts of Kurento technologies are standards; 4) WWW: Kurento model is based on Web development. The vision of Kurento is to assist developers to have advanced multimedia potentials into WWW and Smartphone applications immediately and simply [16]. GStreamer is a framework based on the pipeline structure for multimedia which connects a large amount of media processing systems to include in complicated workflows. Kurento Media Server places the WebRTC media plane on top of GStreamer media handling application which allows the communications of WebRTC to be performed on the server. The development of applications in Kurento depends on two notions: media elements and media pipelines [13]. Media elements enable recording, mixing, augmenting, and blending for applications. Media pipelines are series of media elements which receive and send WebRTC streams. In media pipelines, one element output can be given as an input to one or several elements [17].

2.1.3. Live555 Media Server

Live555 Media Server is a popular RTSP open-source server application, implemented in C++ in which RTSP, RTP protocols are applied for streaming media and this Media Server is compatible with VLC media player and QuickTime Player [18].

Live555 Media server can be considered as a full RTSP application server which is able to stream different types of media files such as, MPEG-4 Video Basic Stream file, MPEG-1 or 2 Program Stream file, and H.264 Video Basic Stream file, etc. The streams are accepted by standard RTSP/RTP media clients, for example: VLC media player, and Quick Time Player [18]. Live555 Media Server is based on RTSP protocol which supports most types of media files. In addition, there is a RTP encoding module in Live555 Media Server which is in charge of the packaging and transferring of data [19].

In order to run Live555 Media server, the user needs to type live555MediaServer in the terminal. Live555 Media Server also supports a functionality called trick play for MPEG Transport Streams [20].

2.1.4. OpenCV 3.1.0

OpenCV is an open source computer vision library which is used for Real-Time computer vision. OpenCV library is written in C, and C++ and can run on Linux, Windows, and Mac OS X [21].

The most beneficial part of OpenCV is its architecture and memory management. Several modules which are built in OpenCV can help to resolve computer vision issues such as crop images and improve them by changing their illumination, image segmentation, detecting moving objects in videos, shape detection, etc. There are some built-in components in OpenCV that are greatly optimized such as HighGUI, Video, MLL, Core and make it powerful and adaptable to resolve all the computer vision issues. Changing the illumination of the images, discovering shapes in the images, finding moving objects in the videos, identifying recognized objects, and cutting images are some of the applications of OpenCV [22].

The structure of OpenCV is made up of five major sections: 1) The CV part consists of basic image processing and the algorithms of computer vision; 2) MLL is a Machine Learning Library that contains several statistical classifiers and clustering tools; 3) HighGUI consists of functions for saving and loading videos and images; 4) The fourth part is CXCORE which has the basic data structure and content [23].

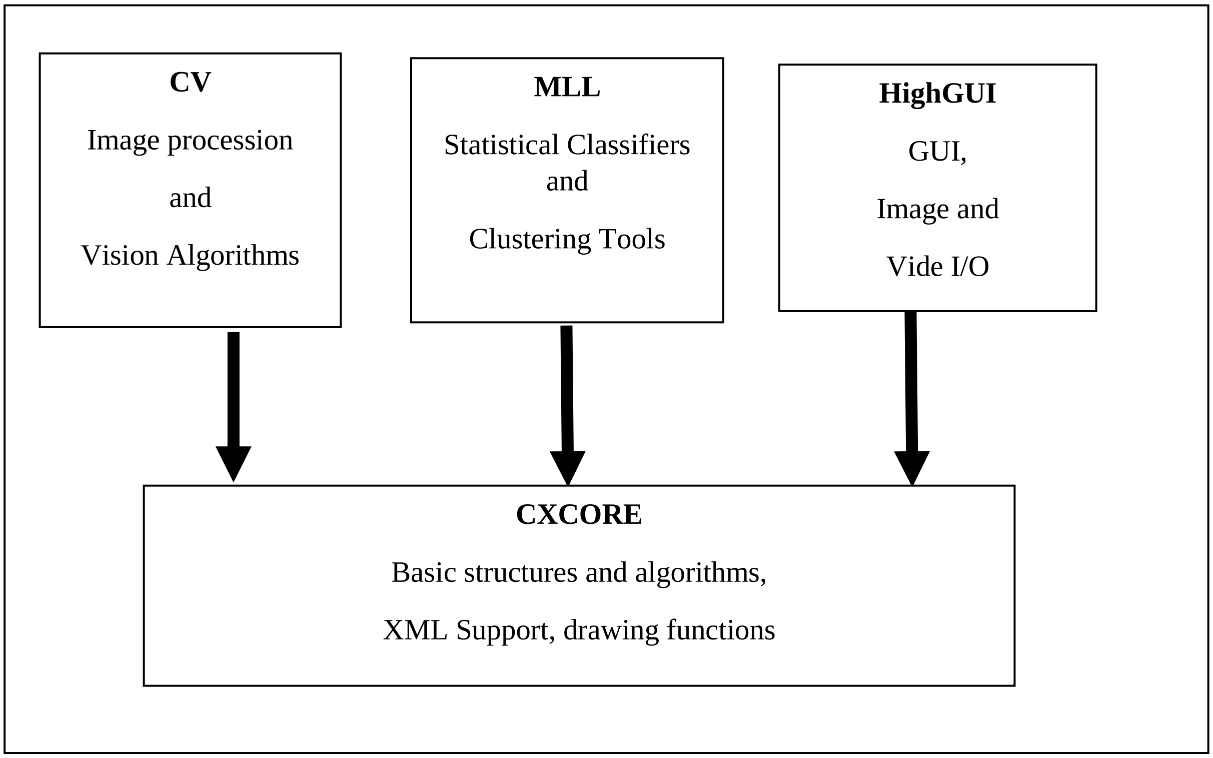


Figure 1: The main structure of OpenCV [23]

OpenCV library supports almost all well-known operating systems such as Windows, Linux, Mac OS, and Android. OpenCV consists of more than 500 optimized algorithms such as machine learning, and computer vision algorithms [24]. OpenCV website and Github are available for downloading its source code. The recent version of OpenCV is 3.1.9 which was released in December 2015. OpenCV can be installed on most of the operating systems with any hardware configurations since it is platform independent. All platforms can be used such as Windows, Linux, and MacOSX, but Linux (Ubuntu) is the best option for installing OpenCV. After downloading OpenCV from its website in Ubuntu, you can extract the folder by Archive Manager or by typing this command: tar –xvf. Then update the system by: `sudo apt-get update` [25].

2.1.5. Java 1.8.0

Java is an object-oriented language and its main objective is to run on any hardware and software platforms which makes it platform independent. Java Development kit includes the software and tools which are required to compile, debug and run applications written in Java. JDK also has a complete Java runtime environment that contains Java Virtual Machine, class libraries and extra libraries which are beneficial for developers [26]. Java programming language, Java Virtual Machine and Java platform are three distinct concepts. When the java program compiles the codes written for applications and servlets in java language, the bytecodes are generated that are called Java Virtual Machine (JVM). In addition, Java platform are a predefined group of java classes which comes along with java installation. The syntaxes of Java programming language are like C. The main objectives for designing Java language were, simplicity, powerfulness, and robustness. Java Virtual Machine is the major part of Java installation. Some of the advantages of using Java programming language are as follows: 1) It is platform independent 2) Java language contains security features 3) Java platform is network-centric 4) Java is dynamic and extensible 5) Java performance is superior 6) Java is efficient 7) Java program has internationalization features [27].

Java's portability characteristic is one of the most important features which is obtained by Java Virtual Machine. A program that is written in Java is being compiled into bytecodes which are independent to the machine. The interpretation of bytecodes will be performed by Java Virtual Machine [28]. The architecture of Java includes the following items: 1) Java programming language 2) The file format of Java class 3) APIs which are Java Application Programming Interfaces 4) The Java Virtual Machine. Java Runtime Environment (JRE) contains Java Virtual Machine and the Java platform core classes. The Java APIs have three major platforms: 1) Java 2 Platform, Standard Edition (J2SE); 2) Java 3 Platform, Enterprise Edition (J2EE); 3) Java 2 Platform, Micro Edition (J2ME). To compile and run the java programs, the following commands are used: `$ javac <filename>.java   $ java <filename>` [29].

2.1.6. PHP 5.5.9

PHP is the acronym for Hypertext Preprocessor, however it is also popular by the name "Personal Home Page". PHP is considered as a server-side programming language. The original idea of creating PHP was to perform simple programming on the Web server. PHP was initially generated by Rasmus Lerdorf in 1994 and it became usable for people in 1995. The initial name of PHP which was Personal Home Page is an apt name since this language facilitates creating web pages on the server [30].

Some features of PHP are as follows: flexible, scalable, extensible, stable, etc. For these reasons, this language is being used on the web servers of more than a third of the planet. This language is very user friendly and non-menacing. It is easy to program and debug in PHP language and that makes it very helpful for beginner programmers. PHP is unique in performance which means that PHP is executing the scripts quicker compared to other scripting languages. Moreover, third-party accelerators exist to enhance even more the performance and the time of response. PHP is known as platform independent and can be run on UNIX, Microsoft Windows, Mac OS and OS/2. PHP is very simple to learn and use which makes it significant. Its syntax is straightforward and consistent. Therefore, learning PHP has become effortless for either a professional or a beginner programmer [31].

If PHP is installed and is available on any machine, by typing “php -v” in the command prompt, the version of PHP will be displayed. In fact, some operating systems such as Linux, and a couple of Unix versions come with PHP. However, on the Microsoft Windows, PHP should be installed [30].

In order to run PHP scripts, some system prerequisites are required: 1) A web server software like Apache or Internet Information Server (IIS) should be available on a computer; 2) PHP server module needs to be installed on the same computer. Since PHP performs very well on Linux operating system and Apache web server, many developers work on Linux to perform PHP programming. PHP can run easily on Ubuntu desktop edition [32].

A set of open source software which is required for a server to host dynamic websites is called LAMP. LAMP is an acronym for Linux operating system, the Apache web server, MySQL database for keeping data and finally PHP [32].

2.1.7. JavaScript

JavaScript is a scripting language which was initially called LiveScript and created by Brendan Eich at Netscape in 1995. This programming language is platform independent and its syntax is like C, Perl, and Java. The JavaScript interpreter is generally built into the browser which executes the JavaScript programs. These programs are utilized to identify and respond to the events caused by the user such as a mouse click or the appearance of a graphic [33].

JavaScript is an object-based programming language meaning that it applies units called objects and contains several features which make it independent to any specific browsers [34]. A JavaScript variable can support several types such as Undefined, Null, Boolean, Number, String, Symbol, and Object. HTML documents and web browsers include JavaScript as well as Document Object Model (DOM) to enable JavaScript to control HTML. DOM supports interfaces for JavaScript to change the content of HTML. The JavaScript Event Handler provides users the ability to interact with the web applications. These Event Handlers can be entered by the developers dynamically by using addEventListener as well as statically by putting onClick, onMouseDown, etc. After invoking the Event Handler, the relevant event is called, and more Event Handlers are invoked in the program [35].

Some of the features of JavaScript are as follows: 1) Case sensitivity: the keywords, variables, function names, and other JavaScript's Syntaxes must be typed following capitalization rules; 2) Similarity with HTML: some of JavaScript properties are the same as the names of HTML tags and attributes; 3) Naming identifiers: the rules for naming the JavaScript identifiers are the same as naming identifiers in Java; 4) Comment styles: JavaScript support both C and C++ styles for commenting; 5) Simplicity along with complexity: JavaScript is a fully-featured programming languages which makes it complex yet simple to understand [36].

2.1.8. K-Means Clustering

One of the method for finding hidden models in data sets is called clustering. In clustering, similar data are categorized in disconnected clusters. One of the most popular clustering algorithms is K-Means clustering. K-Means clustering is a quick and fast clustering method based on grouping the dataset into K clusters in which there exists a centroid that can change flexibly. K-Means clustering is extensively applied in image processing, pattern recognition and data analysis. However, this method has some drawbacks: 1) The number of clusters should be preliminarily defined and is not changeable; 2) The output of this method is widely relevant to the initial centers of the clusters; 3) The algorithm has a dead-unit issue [37] which means if a centroid is improperly selected, it never gets updated and therefore, it cannot illustrate a cluster [38].

One of the restrictions of the K-Means algorithm is that the number of clusters should be pre-defined. Selecting the right number of clusters is very important. Moreover, selecting primary cluster centers or primary seed points can affect the performance of the K-Means clustering [37].

In 1967, MacQueen [39] initially presented K-Means, an unsupervised learning algorithm when unlabeled data is provided. The purpose of the K-Means clustering is to group M points in N dimensions into K clusters [40]. The optimum method in K-Means clustering is to put the centroids as far as possible from each other [41].

The outputs of the K-Means clustering algorithm are 1) labeled new data, for which the centers of the K clusters are used; 2) Every data point is assigned to an individual cluster and the labels can be applied for training data sets [42].

The K-Means algorithm discovers the best centroids by 1) Placing the data points in clusters based on the existing cluster centers and; 2) Modifying the centroids based on placing data points in clusters. In the K-means algorithm, an input dataset is given x_1, x_2,…, x_m to group them into k clusters and the objective of this algorithm is to find k centroids with the label c_i as described in the following steps: 1) Cluster centers are randomly initialized; 2) Data points are assigned to each cluster based on their distance to the centroid of that cluster, for each i, c_i= arg min$||x_\mathrm{i} - \mu_\mathrm{i}||^{\mathbf{2}}$; 3) Centroids are updated by calculating the mean of the points in these clusters until convergence happens, which means no new data points are assigned to clusters. Assigning data points to each cluster depends on the Euclidean Distance of data point and the cluster centers. Updating the centroid of each cluster after assigning every single data point is necessary and is based on the mean of the data points in that cluster. μ_i is randomly assigned as cluster centroids and the Euclidean distances of each incoming points from the centroids μ_i will be calculated [43].

The Pseudo-Code for K-Means Clustering

INPUT: X = {x_1,x_2,x_3,……..,x_n} be the set of data points and V = {v1,v2,…….,vc} be the set of centroids.

```
1. Randomly choose ‘c’ cluster centroids
2. Measure the Euclidean distance between each data points and cluster centroid
3. Assign the data point to its nearest centroid
```

```
4. Recalculate the new cluster centroid by $v_i = (1/ c_i) \sum_{j=1}^{ci} x_i$ ( $C_i$ is the number of data points in $i^{th}$ cluster)
5. Recalculate the Euclidean distance between new centroid and each data point
6. If convergence occurs (no data points can be assigned to clusters),
7. Stop
8. Else, Go to step 2
```

K-Means algorithm is very simple to implement and performs well in most of the practical problems especially when the generated clusters are complicated and have hyper spherical shapes. The K-Means clustering is a good option for clustering large-scale of data sets, since its time complexity is O (NKdT) where T is the number of iterations until convergence takes place, K is the number of clusters, N is the number of vectors, and d is the number of features. One of the issues of the K-Means clustering is that the convergence is not global optimum but on the other hand, K-Means is able to converge to a local optimum. Local optimum is the optimal solution in the neighboring set of all the solutions, compared to global optimum, which is the optimal solution among all possible solutions. Another issue of K-Means is that the number of clusters should be pre-determined by the user which is usually difficult to perform in practice [44].

Some approaches have been examined to resolve the issue of choosing the appropriate cluster number such as invoking heuristic methods. These methods run the clustering algorithm from an initial number of clusters at the beginning, steadily increasing to a particular threshold [37].

The distance between each data point and its closest centroid is sometimes called squared-error. The main goal of K-Means clustering is to decrease squared-error criterion. This process keeps happening until convergence occurs meaning that the squared-error cannot be reduced any more [45].

The K-Means clustering can be performed either in batch mode or in incremental mode. When K-Means clustering is implemented in batch mode which is also called generalized Lloyd algorithm [46], the entire training data is obtainable. On the other hand, K-Means clustering performing in incremental mode is suitable for training sets which can be achieved online [47].

The user must specify three parameters for K-Means algorithms: 1) Numbers of clusters, k; 2) Initialization of cluster; 3) Distance metric. Among these three parameters, the number of clusters, k is the most important one as some heuristic approaches can be applied for this purpose [48]. As mentioned before, the main objective of the K-Means clustering algorithm is to decrease the sum of squared-error in all k clusters. The squared-error is the Euclidean distance of each data point from the centroid of current cluster. The K-Means clustering is known as partitional algorithm clustering in which all the clusters are found at the same time and doesn't contain hierarchical structure [49]. K-Means clustering algorithm is considered as a greedy algorithm and can only reach a local minimum. However, if clusters are properly separated, the K-Means algorithm can reach global optimum as well [50].

2.1.9. Support Vector Machines (SVM)

Support Vector Machines (SVM) were initially developed by Vapnik et al at AT&T Bell laboratories [51]. SVM are used to solve Quadratic Problems (QP) in which the number of instances is equal to the number of variables. Support Vector Machines are considered for a linear and a non-linear classifier. In case of linear classifiers, given data points should be discovered as linearly separable or non-linearly separable. The main objective in all previously mentioned cases is to look for the best hyperplane that separates the data [52].

A binary classification can be given as an easy problem for Support Vector Machines. Support Vector Machines are a supervised learning approach which learn from a training data set and tries to generalize it on a test data set. training data set contains input vectors and for each input vector, there is a label. Considering two classes of distinct data, SVM looks for a directed hyperplane in which the data points on one side are labeled $y_i = -1$ and the data points on the other side are labeled $y_i = +1$. The data points which are nearest to the hyperplane have the greatest effect on the location of the hyperplane and thus, they are called support vectors. The hyperplane can be denoted as w**.** x + b = 0 when **.** is scalar product, b is the bias or offset of the hyperplane from its initial position in input space, x are the points located in the hyperplane and the weights, w are their orientations [53].

Support Vector Machines can be considered for multiclass problems as well, either by combining some binary classifications or by examining the data in an individual optimization formulation. If the dataset is not linearly separable, then the data points are mapped to a feature space where they are separable [54]. A nonlinear map $\emptyset : R^n \rightarrow$ H is examined, and data points are mapped by $\emptyset$ to

H. Instead of the scalar product in SVM for binary classification, the scalar product can be replaced by a kernel which is a function with the following features: K(x,y) = $\emptyset(x).\emptyset(y),$ x,y ∈ R^n. If Support Vector Machines are applied for Multiclass problems, several two-class classifiers are used to resolve the tasks. For instance, the one-against-all approach creates k classifiers. Thus, if ith SVM is being executed, the training data points which have the label i are considered positive and the rest are considered negative. The one-against-one makes k(k-1)/2 SVMs. The ith SVM is trained on two classes, i and j. where the data points that have the label i are considered positive and the data points which belong to j class are considered negative [55].

Kernel Transformation Function is applied to map the input data to a feature space which is a dimensional space. Several kinds of kernels exist to map the input data into different dimensions. If the data are not completely dividable by the Kernel Transformation function, a variable which is called the slack error is utilized to make a soft margin for separating the data. One of the common kernels which is applied in Support Vector Machines, is Gaussian RBF (Radial Basis Function) [56].

2.1.10. J48 Decision Tree

J48 is a decision tree based on the C4.5 algorithm which is an open-source algorithm implemented in java and is available in the Weka data mining tool [57]. J48 classification algorithm builds a decision tree as follows: 1) The tree is made based on top-down recursive concept and training data are all placed in the root of the tree. Attributes are categorical, and training data are recursively divided based on chosen attributes and test attributes are chosen in a heuristic manner; 2) Partitioning the training data will continue until either all the samples of a given node are assigned to the same class or no attributes remain for partitioning [58].

J48 builds a binary tree based on the tuples in a database and classifies these tuples [59].

The Pseudo-Code for J48 algorithm

```
INPUT: Training Data: D
1. T = $\phi$ ;
2. Build the root node and label it with dividing attributes
3. Add arc to root node for split predicate and label
```

```
4. For every arc: D= Database built by utilizing splitting predicate to D;
5. If stop criteria is met, then T= make leaf node and suitable class will be its label.
6. Otherwise, T’ = DTBUILD(D); T= add T’ to arc
```

OUTPUT: Decision Tree: T

When J48 Decision Tree classifies a new item, it builds a decision tree based on the attributes of the training data [59]. The Weka data mining tool provides classification by using J48 Decision Tree as well as several options for tree pruning [57].

One of the disadvantages of J48 Decision Tree is that, its computational complexity depends on the tree depth and should not be greater than the number of the attributes. The depth of the tree depends on the tree size [60]. To execute J48 in the Weka data mining tool, dataset should be created in ARFF format and then Classify tab is selected. J48 can be chosen from the list of the trees [61].

2.1.11. PART

PART is a partial decision tree algorithm which is the combination version of C4.5 and RIPPER (Repeated Incremental Pruning to Produce Error Reduction). The C4.5 algorithm uses a technique which is called divide and conquer, and the RIPPER algorithm extracts the rules using a rule inference approach. The PART algorithm employs the rule inference method to build a group of rules and by utilizing divide and conquer technique it can make partial decision trees [62]. The major difference between PART and these algorithms is its simplicity meaning that when generating proper groups of rules, it doesn’t require the execution of global optimization. The main concept of this algorithm is that it creates a rule and deletes the corresponding instances and keeps generating rules recursively for the instances that are left until none are remaining. This classifier uses the separate-and-conquer approach as well as a pruned tree to generate a rule which enhances its flexibility and speed. In other words, this algorithm makes a partial decision tree which is a normal decision tree that consists of branches to unspecified subtrees. The main advantage of the PART algorithm is its simplicity by integrating the algorithms of rule learning without any need for global optimization. The runtime of the algorithm depends on the number of the rules that it creates, therefore the time complexity is O ($an^2 \log n$) for “n” number of instances in a data set and “a” number of attributes. In general, the number of the rules increases with the size of the training data set [63].

The PART algorithm was originally developed by Frank and Witten and its name was selected due to constantly creating partial decision trees. The decision tree is generated in global optimization and then it is changed to a group of rules and ultimately, it simplifies the rules. In multi-class situations, the ordered list of rules is created which is called a decision list and it is distinct from the usual technique in which an individual rule is made [64]. The given data is evaluated to every rule in the decision list and if any matches are found, the new data is allocated to that group of the rules [65]. Additionally, the method of building rules in PART and RIPPER is different as in PART, an individual rule is related to a leaf whereas, in RIPPER, a rule is made in a greedy way and begins from an empty rule; the conditions are added until the rate of error becomes zero and the procedure is iterated [62].

To create a partial decision tree, a stable sub tree should be found which cannot be simplified anymore and then the algorithm of making a tree is stopped and a rule is extracted [63].

The Pseudo-Code for PART

```
For each given instance, i
Divide i into subsets
Extend the subsets
While (the subsets are leaves and there exist some subsets which are not extended so far)
do (select next subset and extend it)
If all the extended subsets are leaves,
Exchange node, n by leaf
```

2.1.12. Decision Stump

Decision Stump is applied to train the weak classifiers and contains a one-level binary decision tree with categorical attributes [66]. In Decision Stump, the input features which belong to a class are unknown and it can control a large amount of data features. To enhance the accuracy of this classification method, noise and outliers are deleted from the tree. Decision Stump predicts based on the most prominent attribute [67]. Figure 2 below shows how the Decision Stump algorithm works for diagnosing diabetes [68]:

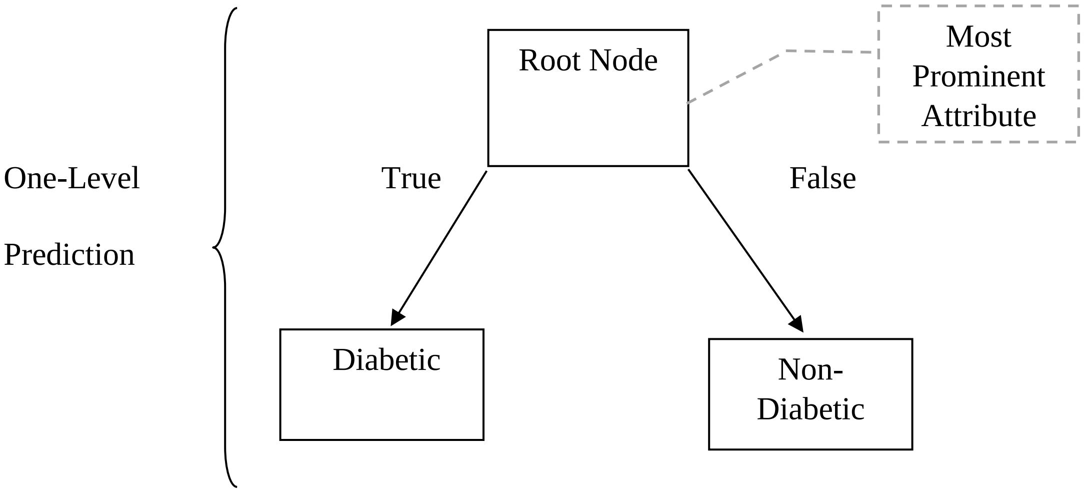


Figure 2: Decision Stump for diagnosing diabetes [68]

Decision Stump classifies by randomly selecting one of the vectors in the input data set which is d-dimensional $x = (x^1, x^2,\dots,x^d)^T$ with a given threshold to that element. One of the disadvantages of the Decision Stump classifier is its accuracy but its computation cost can be considered as an advantage [69]. Labels in this classifier can be either categorical or numerical. Since Decision Stump is a weak classifier it can be integrated with other weak classifiers to build a powerful classifier. One of the algorithms that is used for making a set of classifiers is the Bagging algorithm. This results in a classifier that has a higher accuracy compared with each individual classifier [70]. The Decision Stump classifier algorithm is explained below [71]:

The Pseudo-Code for Decision Stump classifier algorithm

```
INPUT: given pairs $(x_1,y_1)\dots(x_m,y_m)$ ; $x_i \in X$ & $y_i \in \{-1,1\}$ and weights: $D(i) = 1/m$

1. For T=1,…,T
2. return weak classifier: $h_t: X \rightarrow \{-1,1\}$ with minimum error Distribution Dt
3. End for
4. with $\alpha_t \in R$, $\alpha_t$ is the coefficient factor, $D_{t+1}(i) = D_t (i) \exp(\alpha_t y_i h_t(xi))/Z_t$, result is a strong
   classifier, $Z_t$ is normalization factor.

OUTPUT: $H(x) = \text{sign} (\sum \alpha_t ht (x))$
```

2.1.13. Decision Table

There are two types of decision table classifiers which are Decision Table Majority (DTM) and Decision Table Local (DTL). The Decision Table Majority classifier includes two parts which are: 1) Schema, the optimal feature; 2) Body, sets of samples in which each feature and class label has a value. Schema is chosen based on maximizing the cross validation. Labels are assigned to each sample I, if X contains the set of labeled instances that corresponds to I. If X= null, most of the classes in Decision Table are returned. Otherwise, most of the classes in X will be returned. In other words, to label an unlabeled instance, the classifier looks for the same correspondence in the Decision Table by applying solely the schema features and then based on whether the instance is found or not, returning the relevant class [72].

Decision Tables can also be considered as actions and conditions. Class label is called the action and the most corresponding feature is the condition. Each condition is mapped into one action. The technique of cross validation is very necessary for this classifier [73]. Decision Tables can be considered as a specific kind of database relation since the table body is made up of rows or tuples and the Scheme is a group of attributes containing condition and action attributes [74]. Decision conditions need to be analyzed and explained and can be illustrated in a table which is called Decision Table. Thus, data mining information is able to be retrieved from these tables [75]. A Decision Table contains four elements which are separated by each other as shown in Figure 3 below [76]:

Condition subjects	Condition entries
Action subjects	Action entries

Figure 3: Decision Table Diagram [76]

The procedure of decision making is performed by condition subjects. The action subjects explain the possible results of the approach of decision making. The condition entry can have either a corresponding subcategory of values for a particular condition subject or a dash symbol if its value is not related to that column. Moreover, each action entry can have a value related to the action subject class. “x”, “-”, and “.,” are the abbreviations for True, False and unknown action values. Decision Tables have a simple and user-friendly structure in which rules can be retrieved to use [76].

2.1.14. Multilayer Perceptron

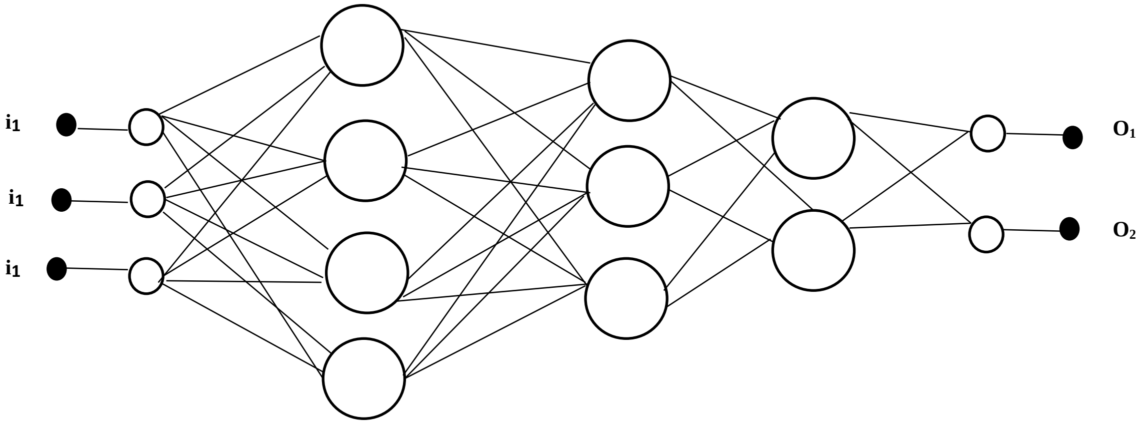
Multilayer Perceptron (MLP) contains nodes which are numerous processing parts connected to each other. An MLP includes one input layer, one output layer and either one or more hidden layers. Every layer has one or more nodes that are depicted by circles. The result of the input layer is received by the hidden layers. The hidden layers transfer the outcome to the output layer [77].

The algorithm of Multilayer Perceptron is explained below [78]:

The Pseudo-Code for Multilayer Perceptron algorithm

```
1. Set the values of weights and thresholds
2. Provide input and desirable output
3. Measure the actual output
4. Adjust weights
5. Begin from output layer and proceeds backwards
6. If the error value of predicted output from actual output is minimized
7. Stop proceeding
```

Figure 4 below displays the architecture of Multilayer Perceptron.



$i = [i_1, i_2, i_3]$ = input vector

$o = [o_1, o_2, o_3]$ = output vector

Figure 4: An MLP with two hidden layers [78]

Multilayer Perceptron is a feed-forward neural network which contains multiple layers of neurons. Input and output vectors have inputs and outputs of Multilayer Perceptron as shown in Figure 4. All the nodes or neurons are interconnected to each node in the adjacent layer along with an activation function allocated to each neuron. Each connection between the neurons has a certain weight meaning that the network applies a Back-Propagation algorithm for tuning the weights. The input vector is fed to the nodes of the input layer and the output vector is the result of the output layer [79]. In each layer, neurons communicate to each other by transmitting the signals through the weighted links. In an MLP, each neuron in the hidden layer contains two functions: 1) Summation Function which computes the sum of every input amount, multiplied by its relevant weight; 2) Activation Function which uses the primary input to specify its signal of the output. In general, the Activation Function is nonlinear, and the Sigmoid Function is the most popular one [80]. The accuracy in an MLP can be affected by several parameters such as the number of nodes in the hidden layer and the number of epochs. An epoch can be defined as a complete training cycle over the training data set. With a lower number of nodes in the hidden layer, a better accuracy can be achieved faster. However, in terms of epochs, each model needs to be examined with a different number of epochs in order to determine a desirable amount which can provide the best value for accuracy [81].

As mentioned earlier, the MLP has a number of layers which consist of a number of neurons. Each neuron is connected to the neurons in the following layer and receives input from the neurons in the previous layer. The neurons connections have carried some weights. By providing enough number of hidden layers, and neurons in each layer of the MLP can obtain the favorable accuracy that is required. Although, the desirable result is only achievable when there is a large amount of training dataset or there is enough number of epochs to let the network to be trained from the training dataset. It is not simple to specify the number of neurons and hidden layers in each MLP as its architecture has a significant role to identify whether it can be trained sufficiently or not. If the architecture is too simple, the network cannot be trained adequately. On the other hand, if the MLP architecture becomes very complicated, it can lead to over fitting [82].

2.1.15. N-Fold Cross-validation

In n-Fold Cross-validation method which is applied for training datasets, n the number of folds is determined. Based on the number of n, the dataset is randomly rearranged and then divided to n equal sizes. In any iteration, one-fold is utilized for test phase and the other n-1 folds are employed for training purposes. The output of test is obtained and averaged through all other folds, then the performance metrics are achieved. The folds can be either randomly created or manipulated to have the same class distributions as the whole dataset which is called stratified [83].

2.1.16. Purity Performance Metric

To evaluate the clustering performance, purity is computed. Purity is an assessment of clustering quality. To measure the purity, after assigning the instances to each cluster, the number of correctly assigned instances are divided by the total number of instances in all the clusters. Purity can be considered as an external evaluation criterion; therefore, the higher the value of purity, the better the clustering output. Purity for each individual cluster is the fraction in which the numerator is the biggest number of instances assigned to that cluster and the denominator is the total size of the cluster. Thus, the final clustering purity is $\sum_{j=1}^{m}\left(\frac{n_j}{n}\right)p_j$, n_j is the size of cluster j, n is the total number of instances, and p_j is the purity calculated for cluster j. The overall purity is obtained by measuring the weighted sum of purity for each individual cluster [84].

2.1.17. Confusion Matrix

Table 1: Confusion Matrix [85]

		Detected	
Actual		Positive	Negative
	Positive	A: True Positive	B: False Negative
	Negative	C: False Positive	D: True Negative

True Positive shows correctly identified instances, whereas True Negative refers to correctly rejected instances. False Positive represents incorrectly identified instances and False Negative refers to incorrectly rejected instances.

2.1.18. Rand Index Performance Metric

Another measure to evaluate clustering performance is Rand Index which is also an external criterion for calculating the percentage of accurate decisions performed by the algorithm. Rand Index refers to how well a clustering method can correctly identify the instances. In other words, Rand Index is an accuracy measure but is applicable when class labels are not applied and can be computed as below based on the given confusion matrix in Table 1 [86]:

$$\mathrm{RI} = \frac{\mathrm{TP+TN}}{\mathrm{TP+FP+TN+FN}}$$

Equation 1: Rand Index

The concept of Rand Index was initially introduced by Rand (1971) and its value is between 0 and 1, where the value 1 means that the instances are correctly clustered, and the output of clustering algorithm matches the ground truth [87]. Ground truth means that clustered manually by human.

2.1.19. Accuracy Performance Metric

In addition, accuracy can be considered as an evaluation criterion for classification approaches. Accuracy is calculated for a given data set based on the percentage of the data which is correctly classified by the total number of predictions. In classification methods, accuracy also represents how well a classifier can correctly classify the instances. The equation below can be employed to compute accuracy of a classifier in which A is True Positive, B is False Negative, C is False Positive, and D is True Negative [58]:

$$\text{Accuracy} = \frac{A+D}{A+B+C+D}$$

Equation 2: Accuracy in Classification Methods

In addition, accuracy can be measured based on True Positive and True Negative values which consider only two classes including positive and negative values, so the denominator of the accuracy fraction contains all the instances as illustrated below:

$$\text{Accuracy} = \frac{TP+TN}{P+N}$$

Equation 3: Accuracy Metric

If the instance is correctly classified, it is TP and if it is correctly rejected, it is TN. P and N are the number of positive and negative values respectively. Accuracy is known as the most important metric for performance evaluation [88].

2.1.20. Recall Performance Metric

Recall is another external evaluation metric which is utilized by both clustering and classification methods. Recall is also called Sensitivity or True Positive Rate (TPR) which reflects how many True Positives are observed. The equation below shows how to calculate Recall [89]:

$$\text{Recall} = \frac{TP}{TP+FN}$$

Equation 4: Recall Metric

The denominator of Recall fraction contains the overall number of correct instances. Recall metric denotes True Positive Rate which represents the ratio of correct instances which are accurately determined. In addition, False Negative Rate can be obtained by the subtraction below [90]:

$$\text{False Negative Rate} = 1 - \text{Sensitivity} = \frac{FN}{TP+FN}$$

Equation 5: False Negative Rate

2.1.21. Precision Performance Metric

Another performance metric is precision or Positive Predictive Value (PPV) which is the fraction of positive instances out of the instances that are truly positive but incorrectly clustered or classified. Precision shows the ratio of correctly observed instances to total number of identified instances. The formula is as follows [89]:

$$\text{Precision} = \frac{TP}{TP+FP}$$

Equation 6: Precision Metric

Precision and recall metrics are extensively utilized for evaluating performance of clustering or classification. Precision denotes the number of instances identified correctly over the total number of instances identified correctly and incorrectly in each class [90].

2.1.22. Specificity Performance Metric

Additionally, specificity is another evaluation metric that is also called True Negative Rate and represents the proportion of incorrect instances which are correctly rejected. Specificity is shown in the equation below:

$$\text{Specificity} = \frac{TN}{TN+FP}$$

Equation 7: Specificity Metric

False Positive Rate shows the probability of falsely rejected instances. False Positive Rate is the ratio of incorrectly identified instances to the total number of negative instances. The ideal value for False Positive Rate is 0 if Specificity is 1. False Positive Rate can be achieved by the subtraction below [88]:

$$\text{False Positive Rate} = 1 - \text{Specificity} = \frac{FP}{FP+TN}$$

Equation 8: False Positive Rate

2.1.23. F-measure Performance Metric

F-measure is another external evaluation criterion, which is a combination or harmonic average of Recall and Precision metrics. F-measure is also called F_1 measure since Recall and Precision are equally weighted. The classical F-measure or F-score can be calculated by using the equation below:

$$\text{F-score} = 2.\frac{Precision.Recall}{Precision+Recall}$$

Equation 9: F-Measure

F-measure is based on a traditional way of retrieving information. For non-negative real values of β, F_β calculates the effectiveness of extraction for β times with the same importance of Precision and Recall where $\beta \in [\ 0, +\ \infty)$. The value of F-measure becomes high only when Recall and Precision values are big [91].

$$F_\beta = (1+\ \beta^2)\ \frac{Precision.Recall}{\beta^2.\ Precision+Recall}$$

Equation 10: F_β -Measure

If β =1 then F_β becomes the harmonic average of Precision and Recall. If β >1 F-measure is Recall-oriented and if β < 1, F-measure gets Precision-oriented. F-measure is commonly used for hierarchical clustering as well as for partitional clustering. Moreover, the data sets with extremely distributed cluster sizes result in significant performance in F-measure [92].

2.1.24. Receiver Operating Characteristics (ROC) Curve

The Receiver Operating Characteristics (ROC) curve is a statistical approach and can be obtained by plotting the values of True Positive Rate (TPR) to the values of False Positive Rate (FPR). This curve was initially used during World War II to analyze and detect the signals and its name comes from the utilization of such curves for signal detection. The word characteristics in this term represents the characteristics of classifier status throughout its operation. This technique was later applied for decision making, radiology, cardiology and nowadays, various fields such as finance, machine learning and data mining utilize this method. The ROC curve illustrates True Positive Rate (Sensitivity) on the vertical axis and False Positive Rate (1 – Specificity) on the horizontal axis and can be considered as a classification performance [93]. The Area Under the ROC Curve (AUC) represents the quality of the classification method and it is used for evaluating the performance of the classifier. It’s worth mentioning that the larger the AUC, the better the classifier. Moreover, to assess the curve completely, its district feature should be selected which is the AUC [94]. The confusion matrix can be applied to make the points in ROC curve. This curve is built to analyze the performance of machine learning algorithms by using some specific datasets.

Every dataset contains the positive and negative instances which can be displays as a point in the ROC curve [95].

2.2. Related Works

Video surveillance systems are designed based on RTSP streaming protocol and H.264, MPEG-4, and MJPEG formats. These systems are able to encode and decode the video frames by using H.264 video codec standard which provides better quality in video images. Additionally, they stream the video frames between the Multimedia client and server as well as play and playback the video images by applying RTSP protocol [96].

Real Time Streaming Protocol (RTSP) streams the consecutive bytes of audio and video data. RTSP has similar functionalities to HTTP. However, there are also minor differences between RTSP and HTTP protocols. In HTTP, the client issues a request and the server replies to that request but in RTSP both the media client and Media Server can issue requests [97].

Current systems for security cameras [98], [99], [100], [101] have proposed a framework for tracking the moving objects such as people and collecting the motion trajectories. The object trajectories are then analyzed in order to distinguish the normal activities from anomalous ones. In [98], a Harris Detector is used to locate the points of interest. Then, the center of gravity of these interest points is calculated and represented as the trajectory of a moving person. The method implemented in [99] considers the object trajectory as a function of time and measures it by using the state vector which includes the relative position of the object. The proposed technique in [100], calculates the object center by performing Bayesian Kalman Filter With Gaussian Mixture. In [101], the object trajectories are obtained based on a uniform cubic B-spline curve which is parameterized by time and the control points.

In [102], Support Vector Machines (SVM) is applied to detect and analyze the object trajectories and tested both with labeled and unlabeled training datasets to be able to detect the outliers in the training sets without using any test dataset. In addition, SVM has been trained and then tested on the test datasets to detect the outlier trajectories in the test sets.

The proposed method in [103] classifies and clusters the object trajectories in which the clustering approach is able to find the similar patterns in the trajectories and SOM algorithm can be used to learn the trajectory patterns from training datasets and classify them over the test dataset in Real-Time.

Elrefaei et al. [104] developed an Android application for detecting the objects based on color, shape and BRISK features. They proposed object detection algorithm by converting RGB color image into HSV color image and Gaussian filter. They used Circular Hough Transform algorithm for circular shape detection and Douglas-Peucker algorithm for rectangle, triangle and square shape detection.

Medioni et al. [105] developed a system that input the video stream and analyzed the behavior of moving objects. A polygon is outlined around the target object in the first video frame and is propagated to the rest of the frames. This system has two modules. The first module detects and tracks moving regions in sequence that uses the set of features at multiple scale for image sequence then extracts the regions and uses attribute graph representation to represent different frames for their trajectories. The second module input these trajectories and user provided information to identify the behavior of moving objects. The combination of spatial and temporal properties is used for the behavior analysis.

C. Piciarelli & G. L. Foresti [106] proposed the trajectory clustering algorithm for the events detection in video surveillance system. In this system, first data are accessed from tracking system then clusters are designed. The clusters are represented in tree like structure that represents the relationship between clusters. The process of tree creation is divided into: building a tree, where a tree of the clusters is developed, and tree maintenance, where updates the clusters in a tree. The branches of the tree are labeled with probabilistic information which is used to detect the unusual activities and to make the prediction of future development of trajectories.

Mishra and Saroha in [107] described a survey on different approaches for static and moving object detection and, moving object tracking. The object detection techniques such as background subtraction, optical flow and frame difference are described. It also explained the criteria for static object detection such as similarity in shape and position, similarity in intensity and edges. It also studied the object tracking methods such as point tracking, kernel tracking and color tracking.

Haritaoglu et al. [108] explained W^4 system to detect and track people and observe their behaviors in an outdoor environment. W^4 system is only used for grey scale and is designed for Real-Time detecting and tracking people and the parts of their bodies.

N. Johnson & D. Hogg [109] proposed statistically based model for object trajectories. The temporal trajectories are modeled using neuron to form vector representation of trajectories. Event recognition is determined by attaching semantics to the area of distribution.

In [110] a program is proposed which can either use a video file as an input with its given path in the source code or it can access the Webcam. The Minimum and Maximum values of HSV color space are given for the green color and a buffer is set with maximum length of 64 to store a list which contains the coordinates of the object center when it is detected. Initially, the video frame is converted to HSV color space and the program checks if there is any contour. If a contour is found, then the center of the contour is calculated. The calculated center is appended to the list. In addition, the contrail of the green ball is drawn in the video frames.

Our proposed method is simpler than the above papers as we ignore any changes in the appearance, pose and illumination of the target object in order to use a straightforward color-based approach for object detection. The reason is to focus on detecting the object of interest and generate the object trajectories based on the object movement and feed them into classification and clustering techniques to solve different problems [111].

2.3. Chapter Summary

Chapter 2 reviewed background information and related works in object tracking and movement detection. The K-Means clustering is an unsupervised learning method which has been discussed as an algorithm and implemented in the Developed Web application. The purpose of the K-Means clustering is to group M points in N dimensions into K clusters and each point in its assigned cluster should have the minimum sum of squares [40]. Support Vector Machines, J48 Decision Tree, PART, Decision Stump, Decision Tables, and Multilayer Perceptron are the classifiers which have been implemented in the batch mode but among them, only SVM classifier was performed in Real-Time Developed Web Application. In addition, several object detection and tracking algorithms have been elaborated. The main difference of this work to previous studies is that, object trajectories which are the series of flow vectors are collected via movement detection and fed into the Developed Web Application to perform Real-Time classification or clustering. The Real-Time K-Means clustering groups the object trajectories in a same cluster if they have similar patterns. In Real-Time SVM classifier, first, the classifier is being trained by a labeled data set, a model is extracted and then the model is applied on generated online trajectories to classify them.

Chapter 3: Methodology

3.1. Implementation of K-Means clustering in Java Program

The K-Means Clustering has been implemented and programmed in Java language from scratch and will be discussed in further detail along with the results in Chapter 4, section 4.3.3.3. The implementation of other classification methods has been performed by using Weka libraries [112].

The Pseudo-Code for K-Means Clustering Implementation in Java Program

INPUT: Number of Clusters = 3, Number of Data Points (x, y) = 60, A Data set file is given

```
While is not the end of input data set file
    For each $i \in \{1, 2, \ldots, 60\}$ in each line of the input file
        $x_i$ is stored in array x, and $y_i$ is stored in array y
    End For
    Array x, and array y are saved in Point p
    Point is stored in a list called allPoints
End While
For each cluster, initial centroids are created
End For
For each cluster i, $i \in \{1,2,3\}$
    For each Point stored in the list called allPoints
        Euclidean Distance of Point and cluster center is calculated and stored in an array
        If the Euclidean Distance of each Point j, $j \in \{1,2,\ldots,60\}$ to each cluster center i, $i \in \{1,2,3\}$is less than the Euclidean Distance of  Point j to other cluster centers
            Assign Point j, $j \in \{1,2,\ldots,60\}$ , to Cluster i, $i \in \{1,2,3\}$
        Cluster centroid i, $i \in \{1,2,3\}$is updated to the mean of the Points in cluster j, $j \in \{1,2,\ldots,60\}$
    End For
End For
If there are no Points left to be assigned to each cluster, or the Iteration of K-Means clustering has reached its Maximum value
    Stop
```

OUTPUT: Cluster number, Cluster center and Cluster data points

3.2. Color-Based Object Detection

The diagram below depicts the work flow of the Developed Web Application which is based on detecting the color of target object.

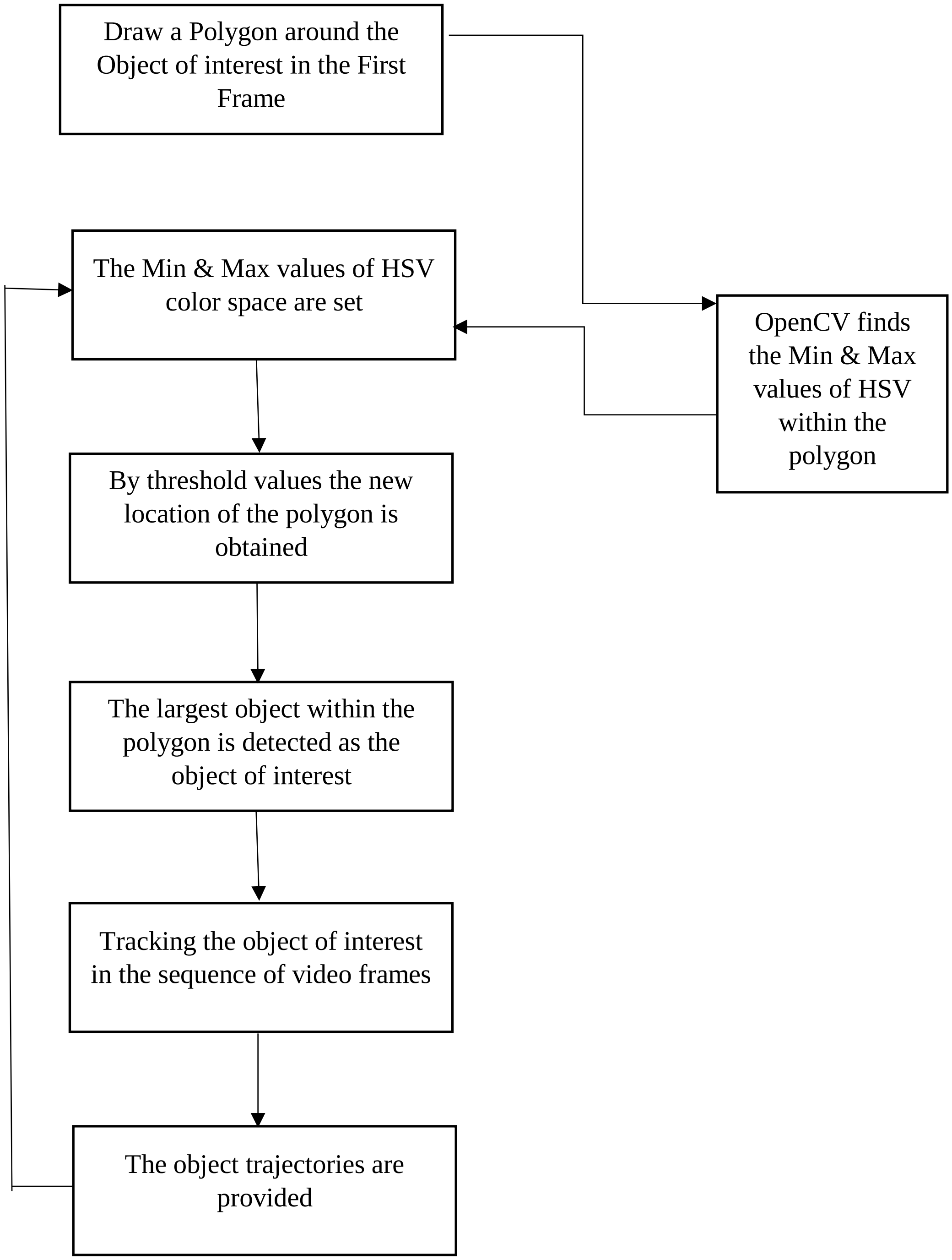

Draw a Polygon around the Object of interest in the First Frame
OpenCV finds the Min & Max values of HSV within the polygon
The Min & Max values of HSV color space are set
By threshold values the new location of the polygon is obtained
The largest object within the polygon is detected as the object of interest
Tracking the object of interest in the sequence of video frames
The object trajectories are provided

Figure 5: The work flow of color-based object detection Algorithm implemented in the Developed Web Application

The algorithm below demonstrates the steps for the color-based object detection approach which has been applied in this thesis:

The Pseudo-Code for color-based Object Detection algorithm

INPUT: The Video Frames

```
1. Read the video file as .mkv format or read the video frames from the IP Camera
2. Initialize a list
3. While (is not the end of video frames in a file)
    1. Convert the frame in RGB color space to HSV color space
    2. Construct a Polygon inside a frame
    3. Find the Min & Max values of H, S, and V of the polygon in HSV color space
    4. Set the lower bound (Min) and upper bound (Max) values of H, S, and V of step 3
    5. Construct a Mask of a frame by giving the Min & Max values of HSV color space // Masking is the process of setting the pixels of background to zero and the object to non-zero
    6. Remove the noise of a mask // Noise are the small spots in the mask
    7. Find the contours in the masks //Contours are the outlines in the mask
    8. For each contour in the mask
        1. Calculate the area of the contour
    9. End For
    10. Find the largest contour
    11. Find the enclosing circle of the largest contour
    12. Find the centroid of the largest contour
    13. Add the centroid to the list
    14. Display the centroid
4. End While
```

OUTPUT: Object Trajectories

3.3. Algorithms for preprocessing Datasets

In this study, there are two main datasets: 1) KnownSource-Destination Dataset; and 2) UnknownSource-Destination Dataset. For preprocessing dataset, two algorithms are applied: Regular Algorithm for KnownSource-Destination dataset and Delta Data Set Algorithm for UnknownSource-Destination dataset.

The Pseudo-Code for Regular Algorithm

```
INPUT: An input data set file with a sequence of coordinates $\{(x_1 , y_1 ), (x_2,y_2),(x_3,y_3),\ldots,(x_{60},y_{60})\}$

While is not the end of the input data set file
    If Number of rows is 150,
        Set i to 150
    Else, set i = 30
    For each i, either $i \in \{1,2,\ldots,150\}$ or $i \in \{1,2,\ldots,30\}$
        For each $j \in \{1,\ldots,60\}$
            Place $(x_j , y_j)$
        End For
    End For
End While

OUTPUT: A sequence of Data set $\{(x_1, y_1), (x_2 ,y_2),(x_3 , y_3),\ldots,(x_{60} , y_{60})\}$
```

In order to apply the UnknownSource-Destination dataset, a novel method is created that is called Delta Data Set Generating method, which is the distance and called delta between each point of data set. Delta of trajectories $(x_1, y_1), (x_2, y_2), (x_3, y_3),\ldots,(x_{60},y_{60})$ can be calculated as ABS (x_i-x_{i-1}) and ABS(y_i-y_{i-1}) based on the following algorithm:

The Pseudo-Code for Generating Delta Data Set Method

```
INPUT: An input data set file with a sequence of coordinates $\{(x_1 , y_1 ), (x_2,y_2),(x_3,y_3),\ldots,(x_{60},y_{60})\}$

While is not the end of the input data set file
    If Number of rows is 150,
        Set i to 150
```

```
Else, set $i = 30$
For each i, either $i \in \{1,2,\ldots,150\}$ or $i \in \{1,2,\ldots,30\}$
Initialize Delta of $(x_1, y_1) = (0, 0)$
For each $j \in \{2,\ldots,60\}$
Delta $_j$ = (ABS $(x_j - x_{j-1})$ , ABS$(y_j - y_{j-1})$)
End For
End For
End While
```

OUTPUT: A sequence of Delta set $\{(0, 0), (x_2 - x_1, y_2 - y_1), (x_3 - x_2, y_3 - y_2), \ldots, (x_{60} - x_{59}, y_{60} - x_{59})\}$

3.4. Performance Metrics for Evaluation of Clustering and Classification Methods

The following performance metrics which are explained in Chapter 2 in detail, are used for evaluating the classification and clustering methods. Rand Index is a clustering performance metric which can be calculated based on the formula below:

$$\text{RI} = \frac{TP+TN}{TP+FP+TN+FN}$$

Equation 11: Rand Index

Table 2: Confusion Matrix [85]

		Detected	
Actual		Positive	Negative
	Positive	A: True Positive	B: False Negative
	Negative	C: False Positive	D: True Negative

In addition, accuracy can be considered as an evaluation criterion for classification approaches. Accuracy is calculated for a given data set based on the percentage of the data which is correctly classified by the total number of predictions.

$$\text{Accuracy} = \frac{A+D}{A+B+C+D}$$

Equation 12: Accuracy in Classification Methods

Moreover, accuracy can be measured based on True Positive and True Negative values which consider only two classes including positive and negative values, so the denominator of the accuracy fraction contains all the instances as illustrated below:

$$\text{Accuracy} = \frac{TP+TN}{P+N}$$

Equation 13: Accuracy Metric

The Recall performance metric is measured as follows:

$$\text{Recall} = \frac{TP}{TP+FN}$$

Equation 14: Recall Metric

Another performance metric is precision or Positive Predictive Value (PPV) which is the fraction of positive instances out of the instances that are truly positive but incorrectly clustered or classified.

$$\text{Precision} = \frac{TP}{TP+FP}$$

Equation 15: Precision Metric

Additionally, specificity is another evaluation metric that is also called True Negative Rate and represents the proportion of incorrect instances which are correctly rejected. Specificity is shown in the equation below:

$$\text{Specificity} = \frac{TN}{TN+FP}$$

Equation 16: Specificity Metric

False Positive Rate shows the probability of falsely rejected instances. False Positive Rate can be achieved by the subtraction below:

$$\text{False Positive Rate} = 1 - \text{Specificity} = \frac{FP}{FP+TN}$$

Equation 17: False Positive Rate

F-measure is another external evaluation criterion, which is a combination or harmonic average of Recall and Precision metrics. The classical F-measure or F-score can be calculated by using the equation below:

$$\text{F-score} = 2.\frac{Precision.Recall}{Precision+Recall}$$

Equation 18: F-Measure

3.5. Receiver Operating Characteristics (ROC) Curve

The Receiver Operating Characteristics (ROC) curve is a statistical approach and can be obtained by plotting the values of True Positive Rate (TPR) to the values of False Positive Rate (FPR). The Area Under the Curve (AUC) shows that an increase in sensitivity will cause a decrease in FP rate. This area can illustrate how well the classifier can determine the currently identified instances and incorrectly identified instances. If the ROC curve is close enough to the top left corner, the value of accuracy that will be gained is higher [113].

3.6. Chapter Summary

Chapter 3 contained the methodology for implementing the Developed Web Application. The Pseudo-Code of the K-Means clustering algorithm implemented in Java program has been discussed. Moreover, the Pseudo-Code of the Color-Based Object Detection algorithm as well as Generating Delta Data Set algorithm have been explained. In addition, the process flow diagram of the Developed Web Application which is based on detecting the color of the target object is described. Finally, the performance metrics such as Accuracy, Rand Index, Recall, Precision, Specificity, Sensitivity, False Negative Rate, False Positive Rate, F-measure, and ROC curve for evaluating clustering and classification techniques have been elaborated.

Chapter 4: Implementations and Experimental Results

4.1. System Configuration

The Developed Web Application has been implemented on the Ubuntu operating system as a result of using the Kurento Media Server. The Kurento Media Server converts RTSP protocol to WebRTC protocol. Its main objective is to present Real-Time communications on the WWW browsers so that users have access to Internet services. The video frames that are captured by a security surveillance camera usually have H.264, MPEG-4, and MJPEG format, which are supported by Live555 Streaming Media Server. By using the OpenCV, which is installed on the Ubuntu OS, the target object is detected based on its color, regardless of any illumination variations. By tracking the object of interest, the video frames are collected and analyzed to obtain the raw image data, which includes the target object's movements. These movements are fed into another model for movement detection. The System configuration for implementation of the Developed Web Application is outlined below:

In-house Server Configuration:

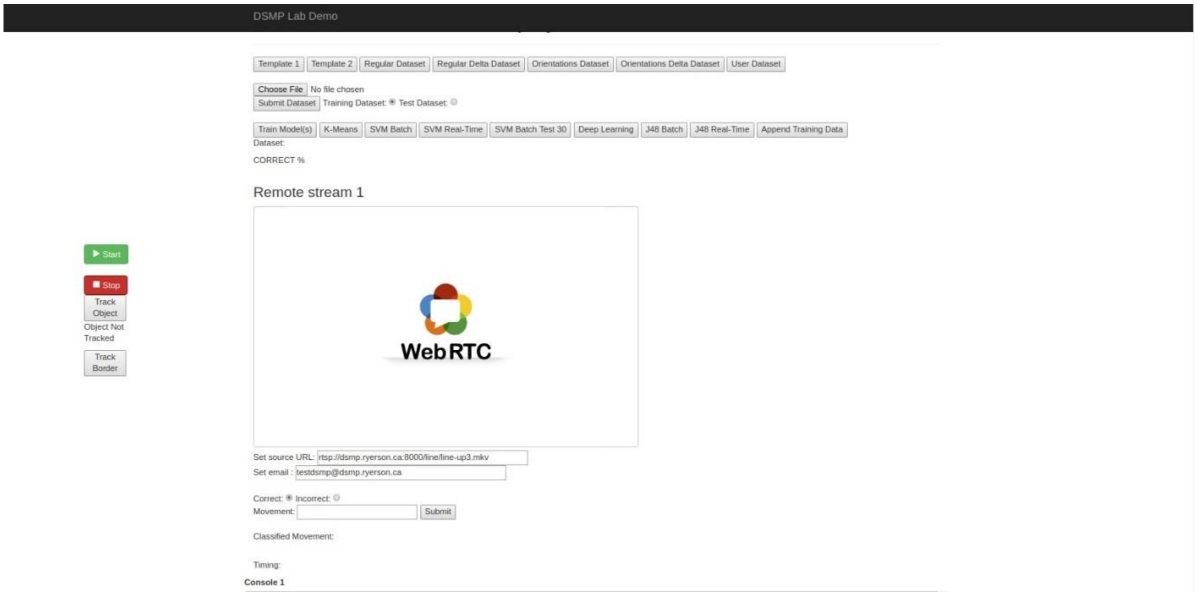
1. Ubuntu Operating System 14.04 LTS
2. Live555 Media Server
3. Kurento Media Server (Release 6.6.1)
4. OpenCV 3.1.0
5. Java 1.8.0
6. PHP 5.5.9
7. JavaScript

Client System Configuration:

1. Internet Browser

4.2. Web Application Implementation

4.2.1. Web Application Implementation for Batch mode



No file chosen

Training Dataset: ● Test Dataset: ○

Dataset:

CORRECT %

Set source URL: rtsp://dsmp.ryerson.ca:8000/line/line-up3.mkv

Set email : testdsmp@dsmp.ryerson.ca

Correct: ● Incorrect: ○

Movement:

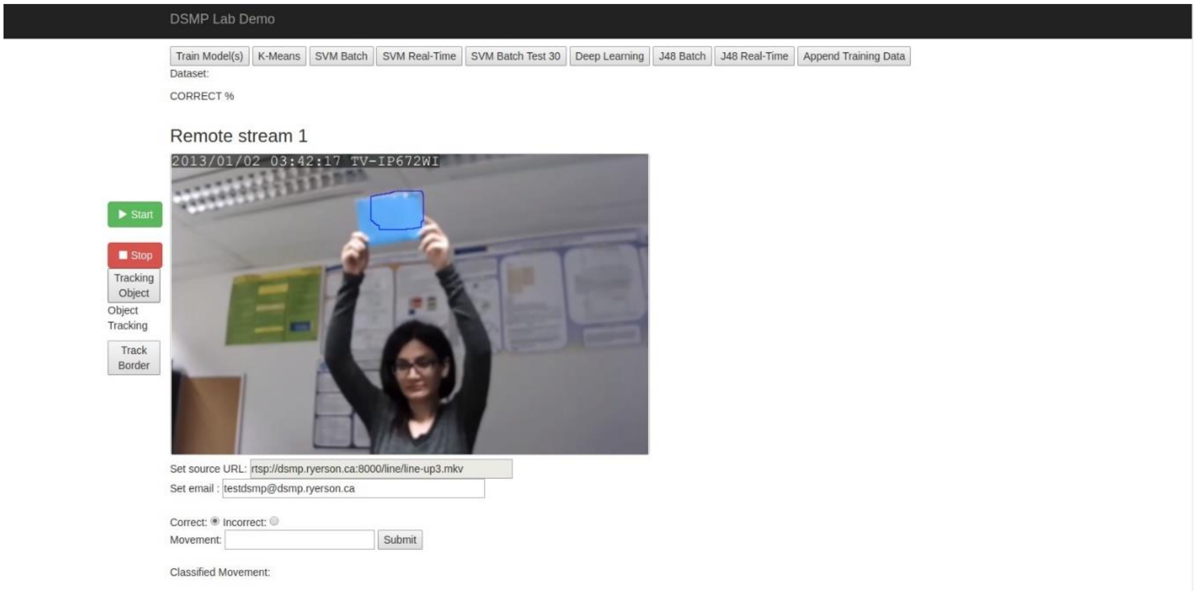
Classified Movement:

Timing:

Figure 6: Screenshot of the Web Application implementation for Batch mode

The URL address of the video file that should be processed, can be input to the Web Application in the Batch mode. The video files that are fed into the Web Application in the Batch mode have .mkv extensions. These files were originally recorded in MP4 format by using the IP camera and the VLC media player. Then, the video files were converted to Matroska with .mkv suffix as Live555 Media Server supports this format. In addition, the Bit rate of the video files that are captured by the IP camera is 64 kbps.

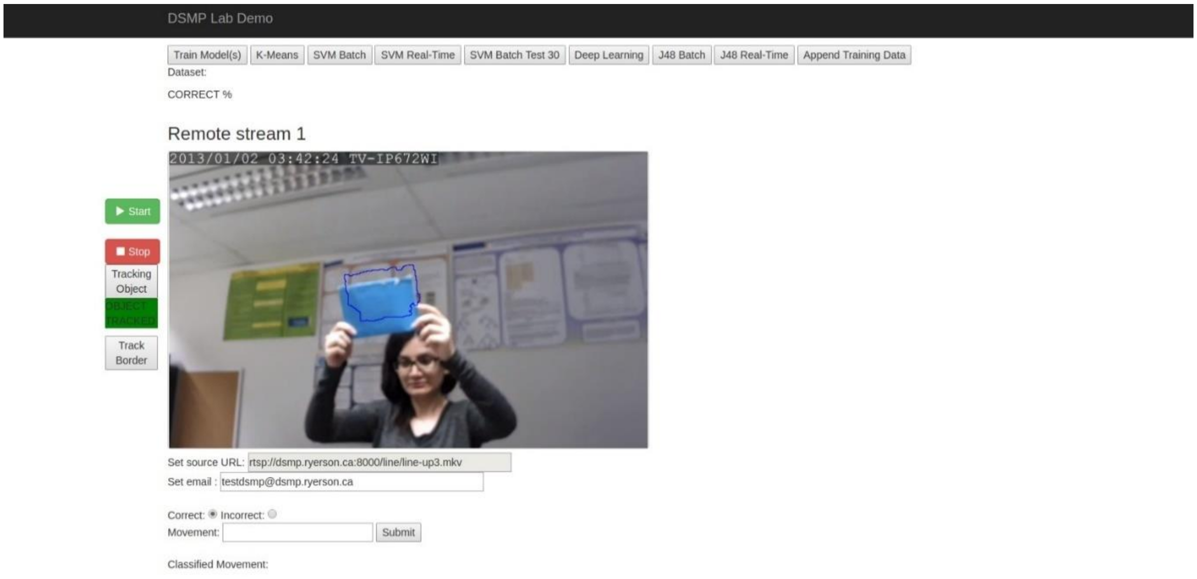
By starting the Web Application, the user draws a polygon around the object of interest in the first frame as it is displayed in the figure 7 below.



Set source URL: rtsp://dsmp.ryerson.ca:8000/line/line-up3.mkv
Set email : testdsmp@dsmp.ryerson.ca
Correct: ● Incorrect: ○
Movement:

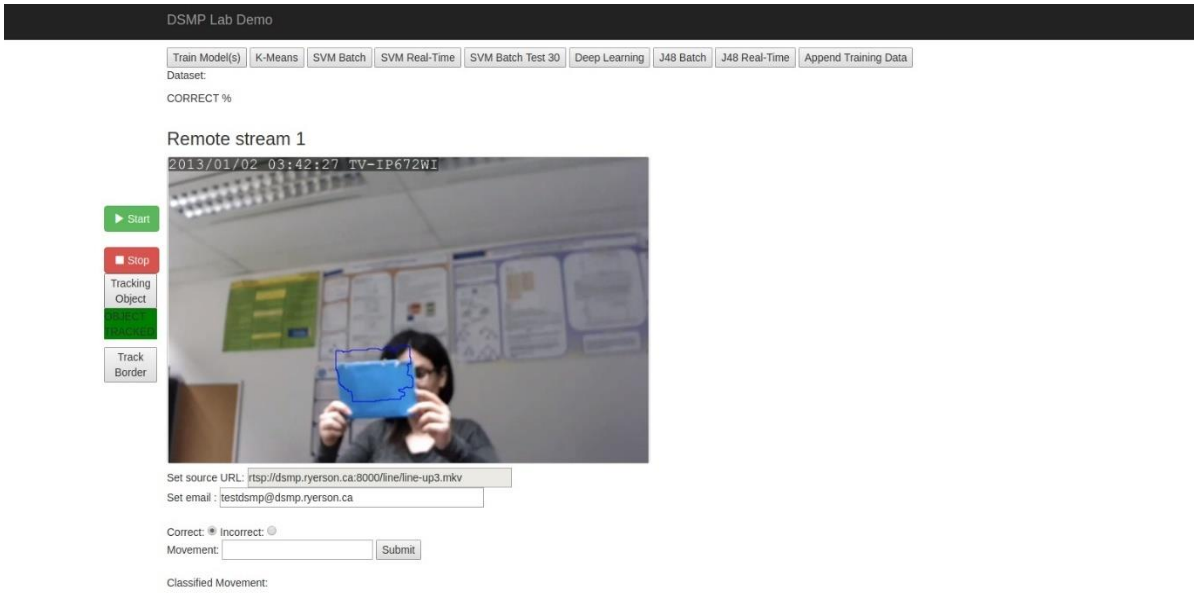
Figure 7: Screenshot of running Web Application in the Batch mode

The Min and Max values of the HSV color space inside the polygon are defined by the Web Application and the object of interest is detected and tracked in the sequence of the video frames.



Set source URL: rtsp://dsmp.ryerson.ca:8000/line/line-up3.mkv
Set email : testdsmp@dsmp.ryerson.ca
Correct: ● Incorrect: ○
Movement:

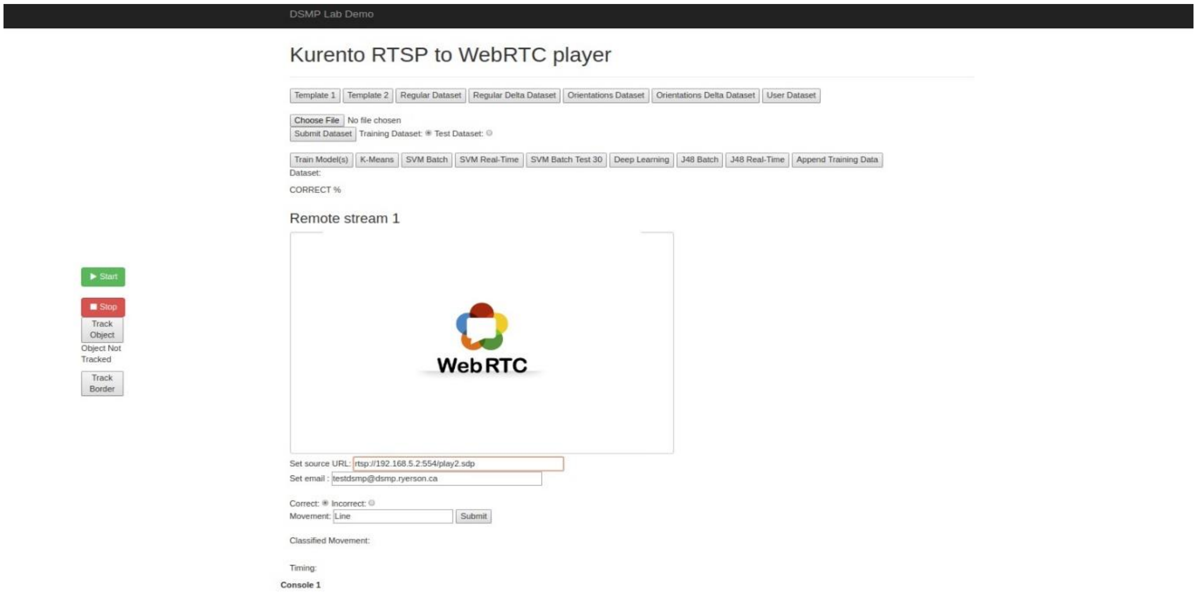
Figure 8: Screenshot of running the Web Application in the Batch mode



Set source URL: rtsp://dsmp.ryerson.ca:8000/line/line-up3.mkv
Set email : testdsmp@dsmp.ryerson.ca
Correct: ● Incorrect: ○
Movement:

Figure 9: Screenshot of running the Web application in the Batch mode

4.2.2. Web Application Implementation for Real-Time mode

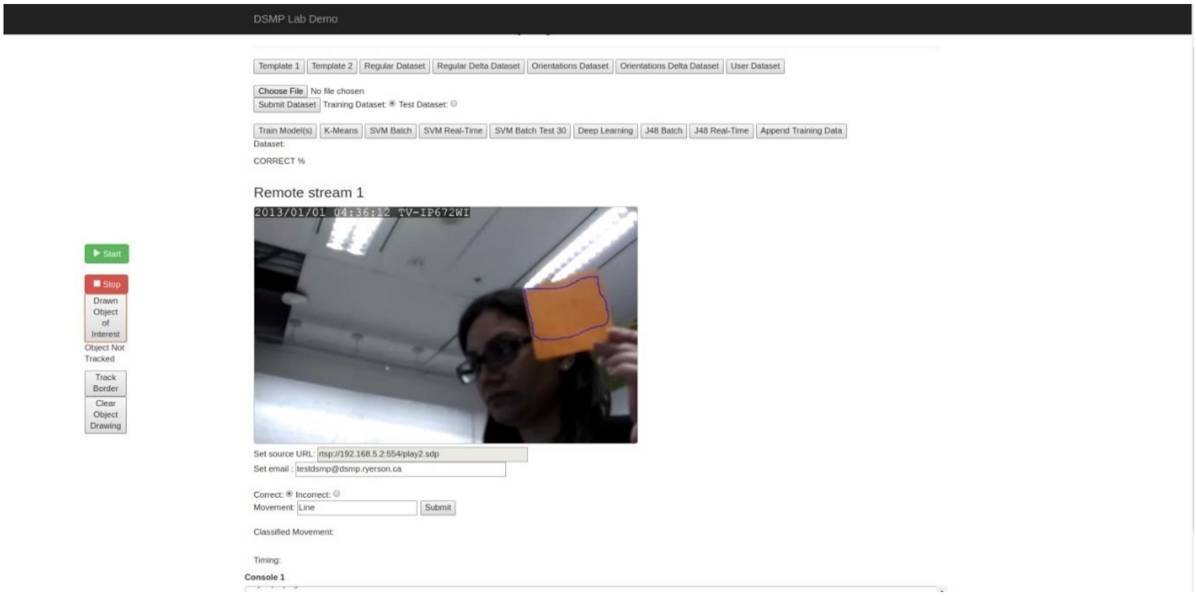


No file chosen
Training Dataset: ● Test Dataset: ○
Set source URL: rtsp://192.168.5.2:554/play2.sdp
Set email : testdsmp@dsmp.ryerson.ca
Correct: ● Incorrect: ○
Movement: Line

Figure 10: Screenshot of the Web Application implementation for Real-Time mode

When the URL address of the IP camera (instead of the URL of the stored video files) is entered in the Web application, it is ready to be used in Real-Time mode. The user should specify which data set is used to train the SVM classifier and press the Train Model(s) button. Once the training is completed, the message, Finished Training Models, will be displayed on the Web Application.

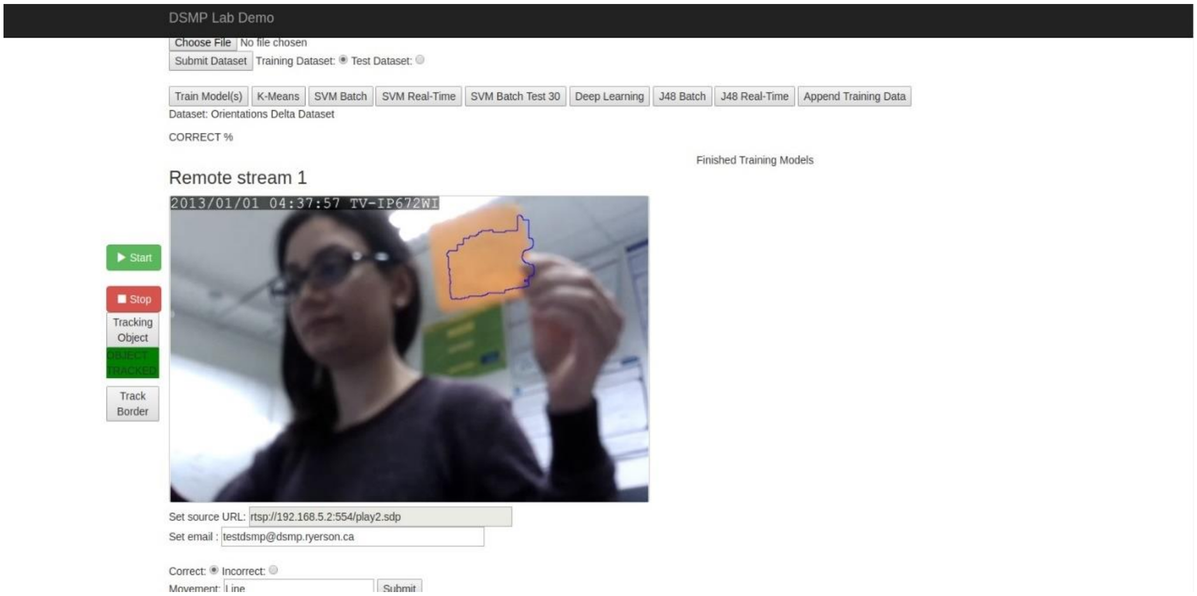
Similar to the Batch mode, in Real-Time, the user draws a polygon around the object of interest to enable the OpenCV built-in functions to find the Min and Max values of HSV color space within the polygon. The figure 11 below illustrates how the polygon has been drawn around the object of interest in Real-Time mode.



No file chosen
Training Dataset: ● Test Dataset: ○
Dataset:
CORRECT %
Object Not Tracked
Set source URL: rtsp://192.168.5.2:554/play2.sdp
Set email : testdsmp@dsmp.ryerson.ca
Correct: ● Incorrect: ○
Movement: Line
Classified Movement:
Timing:

Figure 11: Screenshot of drawing a polygon around the target object by the user in Real-Time

Entering the type of the movement in the Web Application in the Movement field is just a formality and it does not affect the movement detection by the classifier. After detecting and tracking the object of interest in the sequence of video frames in Real-Time, the SVM Real-Time button needs to be clicked.



No file chosen

Training Dataset: ● Test Dataset: ○

Dataset: Orientations Delta Dataset

CORRECT %

Finished Training Models

OBJECT TRACKED

Set source URL: rtsp://192.168.5.2:554/play2.sdp

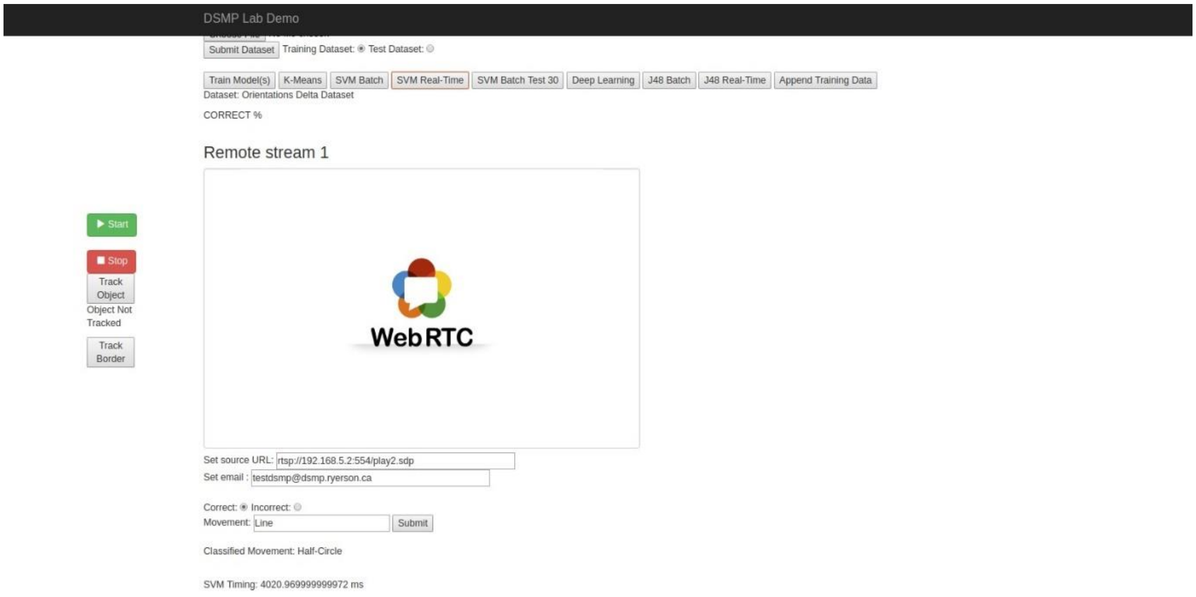
Set email : testdsmp@dsmp.ryerson.ca

Correct: ● Incorrect: ○

Movement: Line

Figure 12: Screenshot of detecting and tracking the target object in the video frames

Regardless of the movement type that has been entered in the Web Application by the user, the SVM classifier detects the movement type and the Web Application shows the name of the movement, accuracy of the classifier, and the timing.



Training Dataset: ● Test Dataset: ○

Dataset: Orientations Delta Dataset

CORRECT %

Object Not Tracked

Set source URL: rtsp://192.168.5.2:554/play2.sdp

Set email : testdsmp@dsmp.ryerson.ca

Correct: ● Incorrect: ○

Movement: Line

Classified Movement: Half-Circle

SVM Timing: 4020.969999999972 ms

Figure 13: Screenshot of the movement detection of SVM Real-Time

4.3. Implementation

The Developed Web Application has been implemented in two modes: Batch, and Real-Time. Developing the Web Application in Real-Time proves that the Real-Time implementation is feasible. The Real-Time Developed Web Application is implemented both on the laptop and our in-house Server which is called DSMP that is used by our research group. In this thesis, we also tested the Developed Web Application on the Ryerson Department of Computer Science cloud environment which is called Cirrus with the configuration of 16 GB RAM, 4 processors, and 40 GB Hard Drive. The results that are presented here is achieved by installing a local host server on a laptop with 12 GB RAM, Intel Core i5 as well as testing the Web Application on the browser of the same laptop. Moreover, TRENDnet TV-IP672WI IP camera is used with up to 30 frames per seconds (fps) as its frame rate and 10/100 Mbps as the Bit rate [114]. In order to be able to install the server on the laptop, VMware Workstation is installed. However, testing the Developed Web Application on laptop has lower running speed, which is the result of several bottlenecks. One of the bottlenecks that affects the speed is utilizing VMware Workstation to run the Web Application on the Ubuntu operating system with the configuration of 12 GB RAM, 4 processors, and 50 GB Hard Drive on Windows operating system. The solution is to install the Developed Web Application on a separate server with Ubunto operating system. If the usage of VMware virtual machine is not required, the installation of the Ubuntu operating system 14.04 LTS which is a Linux distribution on a laptop will also produce a reasonable speed.

4.3.1 Data Sets

The table below illustrates different datasets that were created and used in clustering and classification methods, both in Real-Time and Batch modes. The training datasets include 150 instances, 50 instances of Half-Circle movements, 50 instances of Line movements and 50 instances of Sine movements, whereas Test datasets contain only 30 instances, which are equally 10 instances of Half-Circle, Line and Sine movements.

KnownSource-Destination Datasets, either training or test, consist of object trajectories of the aforementioned movements in one direction with the same origin and destination. This means that all the object movements are created and recorded from the same direction, right to left. UnknownSource-Destination datasets include the object trajectories of these three movements in different directions with different origin and destination. For these two datasets, the data preprocessing methods are applied including: Regular algorithm; and Generating Delta Data Set algorithm, which are explained in Chapter 3 in detail.

The object of interest is tracked by the tracker; therefore, the object trajectories are generated. These are a sequence of coordinates that are {(x_1 ,y_1), (x_2,y_2),(x_3,y_3),…,(x_{60},y_{60})}. Each coordinate (x_i ,y_i) at a given time (t) is called a feature vector. In this thesis, we have 60 feature vectors for both training and test data sets.

Table 3: Data sets in use

Data Set Name	Number of Instances	Number of Features
KnownSource-Destination-Train1	150	120
KnownSource-Destination-Test1	30	120
UnknownSource-Destination-Train2	150	120
UnknownSource-Destination-Test2	30	120

4.3.2. Datasets Collection

In order to provide the mentioned datasets to be used in Batch mode for training and test phased and in Real-Time mode for training purpose, the object of interest is being moved in front of the IP camera to create a specific movement such as, Half-Circle, Line, or Sine and the trajectories are collected after recording this action in a media player. All the datasets contain the three aforesaid movements; Half-Circle, Line, and Sine. The KnownSource-Destination dataset is created by including three movements that are all made in the same direction, from right to left. To create UnknownSource-Destination-Train2 dataset, the movements are generated in different directions as follows: left to right, up to bottom, right to left, and bottom to top as well as diagonal directions. When the sufficient number of trajectories for each movement is collected, the dataset can be used in Batch mode, both in training and test phases, and also for training purpose in Real-Time mode. After training the classifier which is implemented on the Web Application, any movement that is made in Real-Time is considered as test dataset and the target object's movement is detected. Additionally, accuracy metric and running time are obtained by applying performance metrics and measuring the training time for a specific training data set as well as the time that is achieved to detect the type of the movement when the model is generalized over the test data set.

4.3.3. System Implementation in the Batch Mode

4.3.3.1. Results of Classifiers in the Batch Mode by using Weka libraries

In the following table, the results for Multilayer Perceptron is based on 3 hidden layers and “a” number of neurons in each hidden layer. The number of neurons in each hidden layer is calculated by the formula below [115]:

$$a = (\textit{number of attributes} + \textit{number of classes}) / 2$$

Equation 19: The number of neurons in each hidden layer

The performance metrics of all the classifiers below including: SVM, J48, PART, Decision Stump, Decision Table, and Multilayer Perceptron for KnownSource-Destination-Test1 dataset have achieved the accuracy of 100%.

Table 4 displays that Multilayer Perceptron has the Accuracy of 86.67% which is the highest accuracy compared to other classification methods for UnknownSource-Destination-Test2 dataset. It should be noted that, the number of hidden layers in MLP is 3, the number of epochs is 400, and the number of neurons in each hidden layer is “a”.

Table 4: Performance Metrics of all Classifiers in Percentage for UnknownSource-Destination-Test2

	SVM	**J48**	**PART**	**Decision Stump**	**Decision Table**	**Multilayer Perceptron**
Accuracy	73.33	66.67	66.67	50	76.67	86.67
Recall	73.3	66.7	66.7	50	76.6	86.70
Precision	82.1	69.4	67.2	38.6	81.5	88.60
Specificity	86.6	83.3	83.3	75	88.3	93.30
FP Rate	13.3	16.7	16.7	25	11.7	6.7
F-Measure	72.8	66.3	66.5	41	76.2	86.50

The graphs below show the maximum values of Accuracy in percentage for each classifier along with its related data set. The higher the value of accuracy, the better classification methods can correctly identify the instances [88]. The following results are based on 10-fold cross validation on 180 instances of KnownSource-Destination and UnknownSource-Destination datasets.

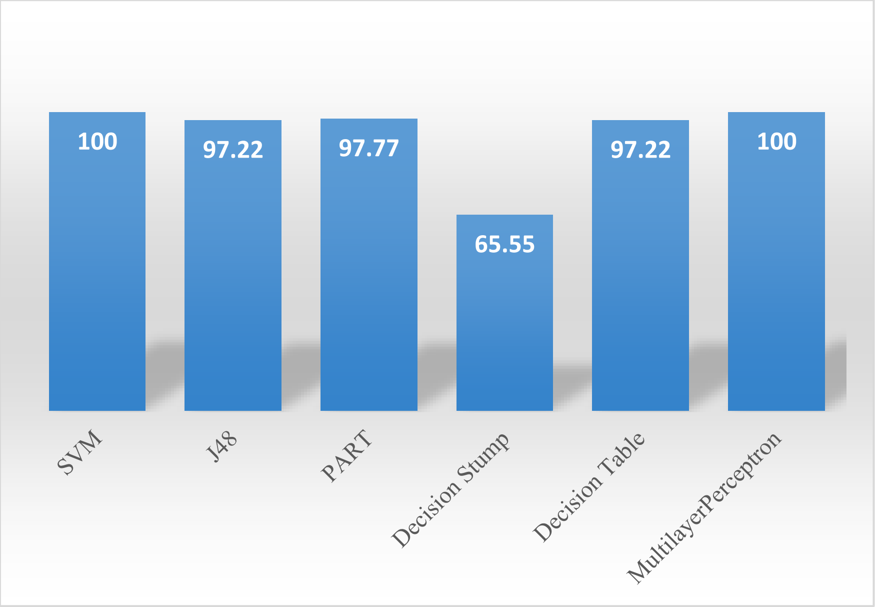


Figure 14: Accuracy Percentage of different classifiers for KnownSource-Destination

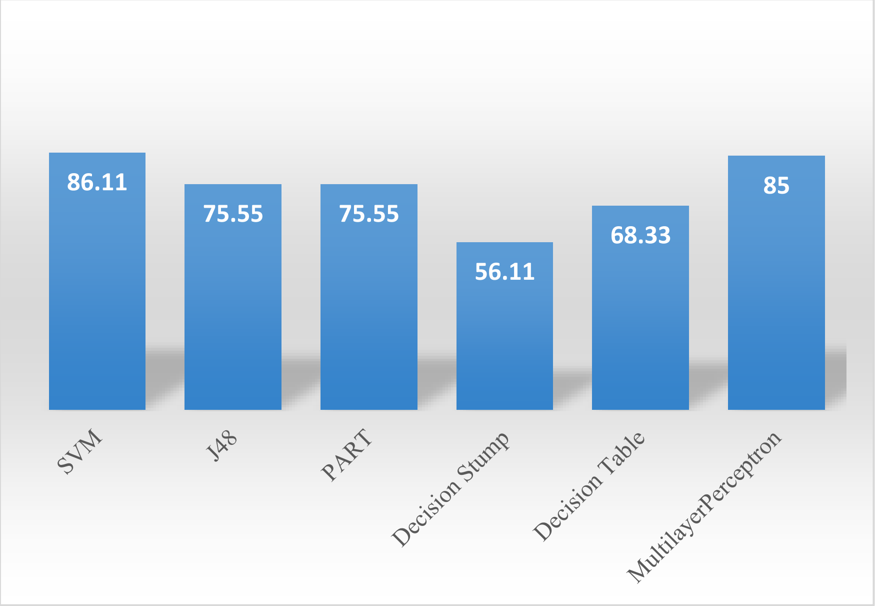


Figure 15: Accuracy Percentage of different classifiers for UnknownSource-Destination

The results show that the SVM and Multilayer Perceptron outperform the other classifiers with regard to the highest amount of accuracy as a performance metric. Among all aforementioned classifiers, the Multilayer Perceptron has achieved the accuracy of 86.67% for UnknownSource-Destination-Test2 dataset in Batch mode. Additionally, SVM and Multilayer Perceptron have obtained the highest accuracy which is 100% for KnownSource-Destination-Test1 based on 10-fold cross validation on 180 instances in Batch mode.

4.3.3.2. Results of K-Means Clustering with Different Datasets in Batch Mode

Table below shows the evaluation metrics for the K-means clustering method. Since the K-means clustering is an unsupervised learning method and does not need to be trained, each training and test dataset were clustered individually.

Table 5: K-means clustering results in Percentage for Datasets in use

Data Set	**Rand Index**	**Recall**	**Precision**	**Specificity**	**FP Rate**	**F-measure**
KnownSource-Destination-Train1	100	100	100	100	0	100
KnownSource-Destination-Test1	97.33	96.7	93.6	98.3	1.7	95.1
UnKnownSource-Destination-Train2	69.70	54.6	51.1	77.3	22.7	52.7
UnKnownSource-Destination-Test2	73.30	60	55.7	80	20	57.7

4.3.3.3. Results of K-Means Clustering implementation without using Weka libraries

To be able to see the effect of using K-Means implementation instead of Weka libraries a separate K-Means program for batch mode has been developed. The table below shows the result of K-Means implementation and evaluation metrics for K-means clustering using different datasets. The results are comparable with using Weka libraries although this time By KnownSource-Destination-Test1 has achieved the highest value of Rand Index among other datasets.

Table 6: K-means clustering results in Percentage for Datasets in use

Data Set	**Rand Index**	**Recall**	**Precision**	**Specificity**	**FP Rate**	**F-measure**
KnownSource-Destination-Train1	99.53	99.3	99.3	99.6	0.4	99.2
KnownSource-Destination-Test1	100	100	100	100	0	100
UnknownSource-Destination-Train2	71.06	57.9	46	78.3	21.7	51.2
UnknownSource-Destination-Test2	68.86	53.3	66.1	76.6	23.4	58.9

4.3.4. Implementation in Real-Time

4.3.4.1. Results of Different Datasets in Real-Time Mode

In this thesis, two Machine Learning methods, SVM classification method and K-Means clustering technique have been implemented in Real-Time in the developed Web Application which is the proof of the concept that Real-Time implementation is feasible.

Tables 7 and 8 illustrate the result of K-means clustering and SVM classification being implemented in Real-Time in the Web Application. The results indicate that by using KnownSource-Destination-Test1, K-means has obtained the Accuracy of 96.67% and SVM classifier has achieved the highest value of Accuracy which is 100%.

Timing evaluates the running time of clustering and classification methods.

Table 7: K-means clustering results for Datasets in use

Data Set	Accuracy (%)	K-means Timing (ms)	Correctly Clustered	Total Instances
KnownSource-Destination-Test1	96.67	2594.92	29	30
UnknownSource-Destination-Test2	60.0	2402.739	18	30

Table 8: SVM Classification Results for Datasets in use

Data Set	Accuracy (%)	SVM Timing (ms)
KnownSource-Destination-Test1	100.00	1487.219
UnknownSource-Destination-Test2	73.33	1874.474

By looking at the timing results of SVM and K-Means clustering it become clear building a system is feasible and in 1 or 2 seconds the movements can be detected in Real-Time.

In Real-Time mode the accuracy is calculated by first training the SVM with 150 instances and then after 30 Real-Time movements, SVM has also gained the accuracy of 100% for KnownSource-Destination-Test1 with 1487.219 ms timing which proves that SVM is faster and more accurate than K-means clustering in Real-Time.

4.4. Chapter Summary

Chapter 4 discussed the Developed Web Application being implemented in two modes: Batch and Real-Time. Developing the Web Application has been performed in Java, JavaScript, and PHP programming language. Java is an object-oriented language and its main objective is to run on any hardware and software platforms which makes it platform independent and PHP is a server-side programming language. There are two different data sets. KnownSource-Destination and UnknownSource-Destination datasets. The training datasets include 150 instances; 50 instances of Half-Circle, 50 instances of Line, and 50 instances of Sine movements. However, Test datasets contain only 30 instances, which are equally 10 instances of Half-Circle, Line, and Sine movements. The results of classification and K-Means clustering methods have been displayed in several tables for comparison purposes.

Chapter 5: Conclusion and Future Studies

5.1. Conclusion

The goal of this research is to develop a Web Application for a Web Video Player presented on a client's machine which communicates with a Video Streaming in-house Server that security surveillance cameras connect to. The Web Video Player identifies the object of interest and retrieves data from it, processes the data and uses the processed data to record the trajectories or send notifications to client's machine (for example sending an alarm in case of anomaly occurrence). With the application of data mining and machine learning techniques such as Support Vector Machines (SVM), J48 Decision Tree, PART, Decision Table, Decision Stump, Multilayer Perceptron classifiers, and K-means clustering, a model is extracted. Two different datasets are used in this study including: KnownSource-Destination Train & Test datasets, and UnknownSource-Destination Train & Test datasets. The method tracks the object of interest among the video frames collected by a security surveillance camera and streamed by a web media server. The video frames are analyzed to collect the movement of the object of interest by obtaining object trajectories and feeding them into the classification or clustering methods for any training or test purposes.

The main objectives of applying Generating Delta Dataset method are to apply different orientations of object movements and achieve higher accuracy in order to test machine learning approaches. These approaches are trained and tested with object movements in different directions where the object is detected by its color. Various popular machine learning methods that are used in this thesis are implemented in Batch mode as well as in Real-Time mode in a Developed Web Application for the comparison purposes.

In this study, Real-Time implementation has become feasible by a Web Application installed on the Ubuntu operating system and is written in JavaScript, PHP and Java programming languages with the streaming of a media server by Live555 open-source Media Library and Kurento Media Server in order to achieve Real-Time performance metrics compared to Machine Learning techniques in Batch mode. In the developed Web Application, the K-Means clustering and SVM classification methods have only been implemented in Real-Time. If any anomalous behavior is detected, the Real-Time tracking system should notify the user by either sending an alarm or making a public announcement. Thus, detecting and tracking a moving object to identify the suspicious activities is one of the objectives of the developed Real-Time Web Application for security purposes.

By implementing the K-Means Clustering method, there is no need for human or domain experts to label the datasets as it is an unsupervised learning technique. The accuracy of SVM classification method is 73.33% and the accuracy of the K-Means clustering is 60% for UnknownSource-Destination-Test2 which means that the accuracy of the supervised learning method is around 13% higher than an unsupervised learning technique in Real-Time. Six different performance metrics have been calculated: Accuracy, Recall, Precision, Specificity, FP Rate, and F-measure.

In conclusion, the proposed developed Real-Time Web Application for multimedia streaming has been implemented to perform Real-Time classification and clustering techniques by using different datasets based on the movement of the object of interest detected by its color. The video frames are captured by a security surveillance camera and streamed by a Web Media in-house Server. The object trajectories are collected based on the movement of the target object and then fed into the Web Application for Real-Time classification and clustering purposes. It should be noted that the object movement can be Linear, Half-Circular or Sine movements as the supervised learning methods have only been trained based on these movements.

5.2. Contributions

The main contributions of this thesis are listed as follows:

- Developed an in-house network architecture to detect and track the object of interest via a web browser and generate the object trajectories in the server side to be classified in Real-Time and detect the movement type that is being captured by using an IP Camera.
- A Web Application was developed to detect and track the object of interest through a web browser by using RTSP/WebRTC protocols to address research question one. This Web Application can be used for Batch and Real-Time data mining for trajectory detection.
- Two data sets were created from hundreds of videos, including three types of movements with known and unknown source and destination. These video files and data sets which contain labels can be used by other researchers for research purposes and are accessible to the public in [6].

- Several Machine Learning approaches for detecting the moving object trajectories in Batch and Real-Time modes are compared. Acquired by observation, the conclusion was to detect the object trajectories of a moving object, one can use PART, J48 Decision Tree, Decision Stump, Decision Table, and Multilayer Perceptron for Batch mode and the K-means clustering and SVM for Real-Time mode in the Web Applications that employed the security surveillance cameras.

5.3. Future Works

Regarding future research directions, the implementation of a Web Application to be able to perform Real-Time machine learning methods, including more supervised and unsupervised learning techniques, can be considered. Additional supervised and unsupervised techniques, aside from SVM classification and K-Means clustering methods, can be implemented in Real-Time Web Application as future works.

Further different datasets from real scenarios can be applied for training and test phases by making more complex movements such as Triangle, Square, or Circle, which are created offline as well as in Real-Time via a security surveillance camera/IP camera connected to a computer. Therefore, detecting more movements can be considered as another future study.

Additionally, another possible future direction can be the object tracking and movement detection not only based on the object's color, as color-based object detection is one of the most common methods in this field. Applying more sophisticated object tracking and object detection algorithms can be noted for expanding this work. The object detection and tracking can be enhanced to take into account more complex movements, multiple moving objects, illumination variations, different frame rates and bit rate in network as well as different background environments.

It is certainly worth conducting future research in developing an event server to notify the user by triggering alerts (e.g. calling the police, closing a door, turning on a light) if any anomaly occurs during object tracking and movement detection by using the clustering and classification methods of the Web Application.

Instead of using an IP camera, applying a built-in camera such as a Webcam in client's machine or a smartphone camera in smartphones as a replacement for the client's machine can be examined as another research direction. Moreover, this system is restricted to be merely implemented on the

Ubuntu operating system; other operating systems such as Windows and MacOS should be included in future research studies by omitting the Kurento Media Server.

Appendix I

Installing GUI for Ubuntu Server

Step1: Performing update.

```
$ sudo apt-get update
```

Step2: Installing the desktop.

```
$ sudo apt-get install ubuntu-desktop
```

Step3: Rebooting.

```
$ sudo reboot
```
[12]

Appendix II

Configuring and building the Live555 Media Server source code

Step1: Running as Root user: `$ sudo su`

Step2: Get the Live555 source code

```
$ wget http://www.live555.com/liveMedia/public/live555-latest.tar.gz
```

Step3: Extract the package: `$ tar -xzf live555-latest.tar.gz`

Step4: In the directory that Live555 is extracted, run: `./genMakefiles`

Step5: Building the code: `$ make`

Step6: Installing the libraries and headers: `$ make install` [20]

Appendix III

Installing OpenCV on Ubuntu 14.04 LTS

```
$ sudo apt-get install build-essential
```

```
$ sudo apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-dev libavformat-dev libswscale-dev
```

```
$ sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev libjpeg-dev libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev
```

The latest version of OpenCV from the Git Repository can be downloaded by:

```
$ git clone https://github.com/Itseez/opencv.git
```

```
$ git clone https://github.com/Itseez/opencv_contrib.git
```

1) Make a directory to put project and object files as well as Makefiles:

```
$ mkdir build  ,  $ cd build
```

2) Run cmake for configuration:

```
$ cmake -D CMAKE_BUILD_TYPE=Release -D CMAKE_INSTALL_PREFIX=/usr/local
```

3) Explanation of some parameters:

```
$ CMAKE_BUILD_TYPE=Release\Debug
```

4) Build:

```
$ make -j7 # runs 7 jobs in parallel
```

5) Library installation:

```
$ sudo make install
```

6) Run the tests:

```
$ git clone https://github.com/Itseez/opencv_extra.git
```

[116]

Appendix IV

Installing Java on Ubuntu Operating System

Step1: `$ sudo apt-get update`

Step2: `$ sudo apt-get install openjdk-8-jdk`

Step3: Verify the installation. `$ java -version` [28]

Appendix V

Installing PHP on Ubuntu Operating System

```
$ sudo apt-get install php libapache2-mod-php php-mcrypt php-mysql
```

The next step is that to make web server searches for `index.php` when a directory is needed:

```
$ sudo nano /etc/apache2/mods-enabled/dir.conf
```

Apache web server needs to be restarted by this command:

`$ sudo systemctl restart apache2` [117]

Appendix VI

Confusion Matrix of different classifiers for UnknownSource-Destination-Test2 Dataset

== SVM Confusion Matrix==	==J48 Confusion Matrix ==	==PARTConfusion Matrix ==
a b c <-- classified as	a b c <-- classified as	a b c <-- classified as
5 4 1 | a = half-circle	5 2 3 | a = half-circle	6 2 2 | a = half-circle
0 10 0 | b = line	0 8 2 | b = line	0 8 2 | b = line
0 3 7 | c = sine	1 2 7 | c = sine	2 2 6 | c = sine

Figure 16: Confusion Matrix of SVM, J48, and PART for UnknownSource-Destination-Test2

== Decision Stump CM ==	== Decision Table CM ==	=MLP CM=
a b c <-- classified as	a b c <-- classified as	a b c <-- classified as
0 9 1 | a = half-circle	7 3 0 | a = half-circle	10 0 0 | a = half-circle
0 9 1 | b = line	0 10 0 | b = line	1 9 0 | b = line
0 4 6 | c = sine	2 2 6 | c = sine	0 3 7 | c = sine

Figure 17: Confusion Matrix of Decision Stump, Decision Table, and MLP for UnknownSource-Destination-Test2

Appendix VII

ROC Curves for Classification Methods

ROC Curve displays False Positive Rate on X axis and True Positive Rate on Y axis. The rate of misclassification can be measured by the distance of the curve and the top left corner of the plot [93]. The Area Under the Curve (AUC) shows that an increase in sensitivity will cause a decrease in FP rate. This area can illustrate how well the classifier can determine the currently identified instances and incorrectly identified instances. If the ROC curve is close enough to the top left corner, the value of accuracy that will be gained is higher [113].

VII.I. ROC Curves for UnknownSource-Destination-Test2

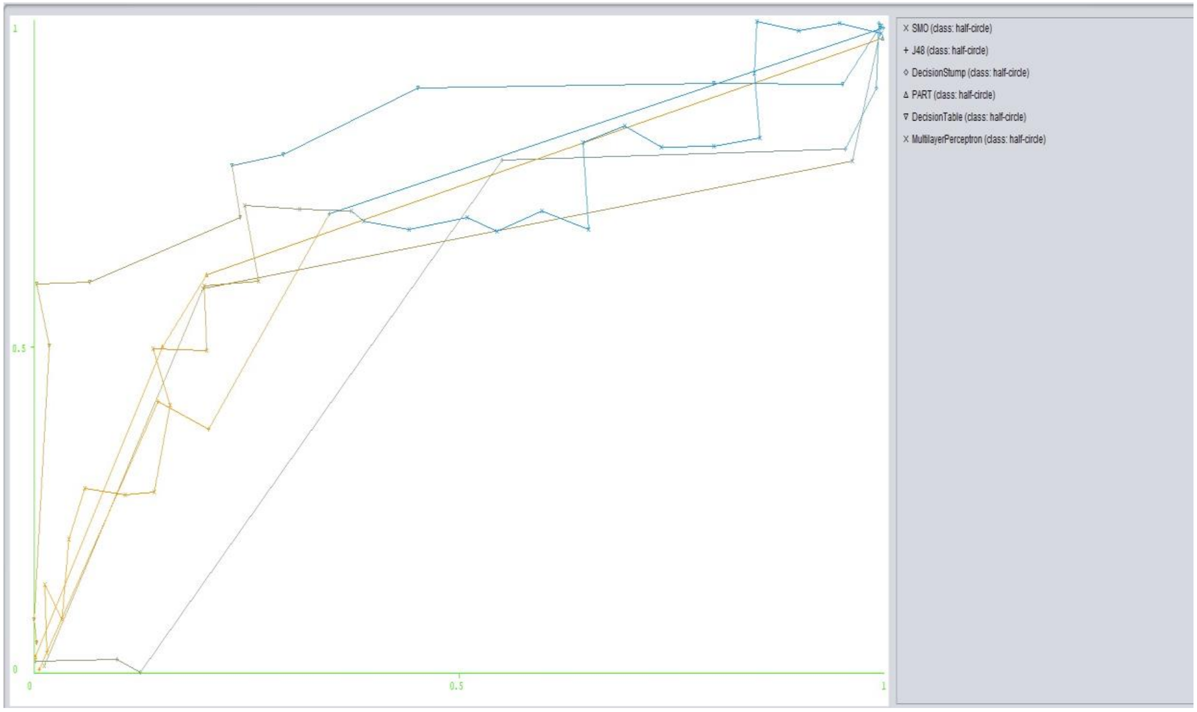


Figure 18: ROC Curve of UnknownSource-Destination-Test2 for class Half-Circle

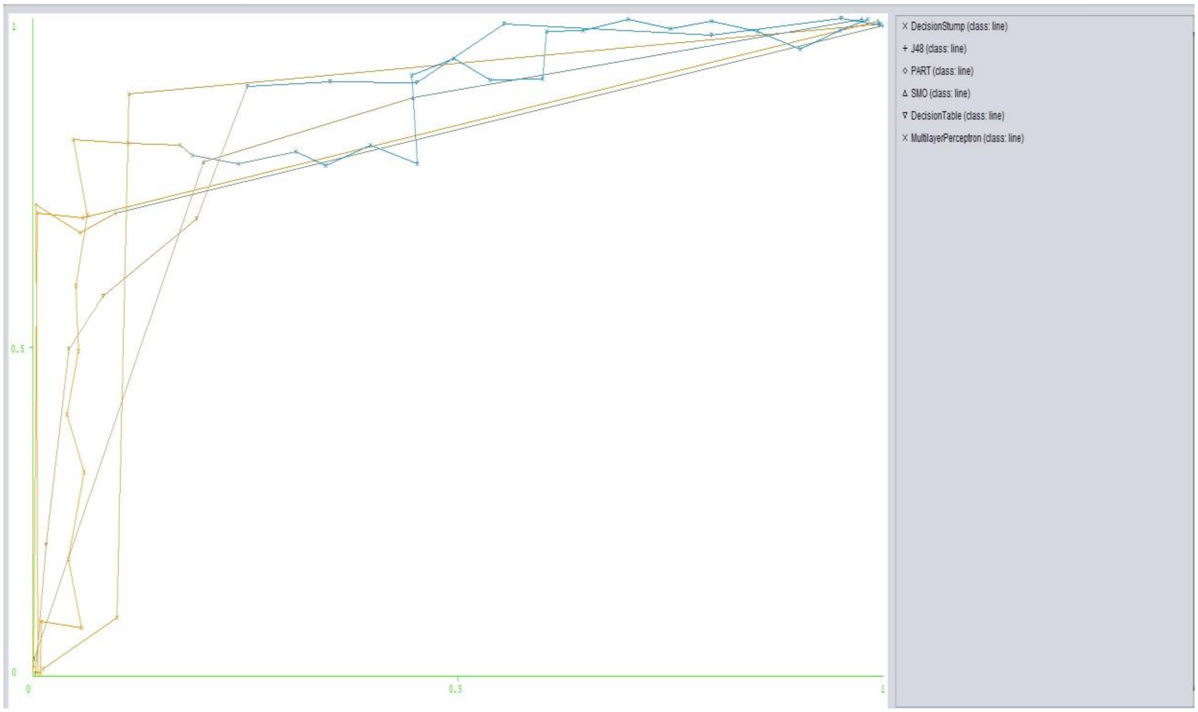


Figure 19: ROC Curve of UnknownSource-Destination-Test2 for class Line

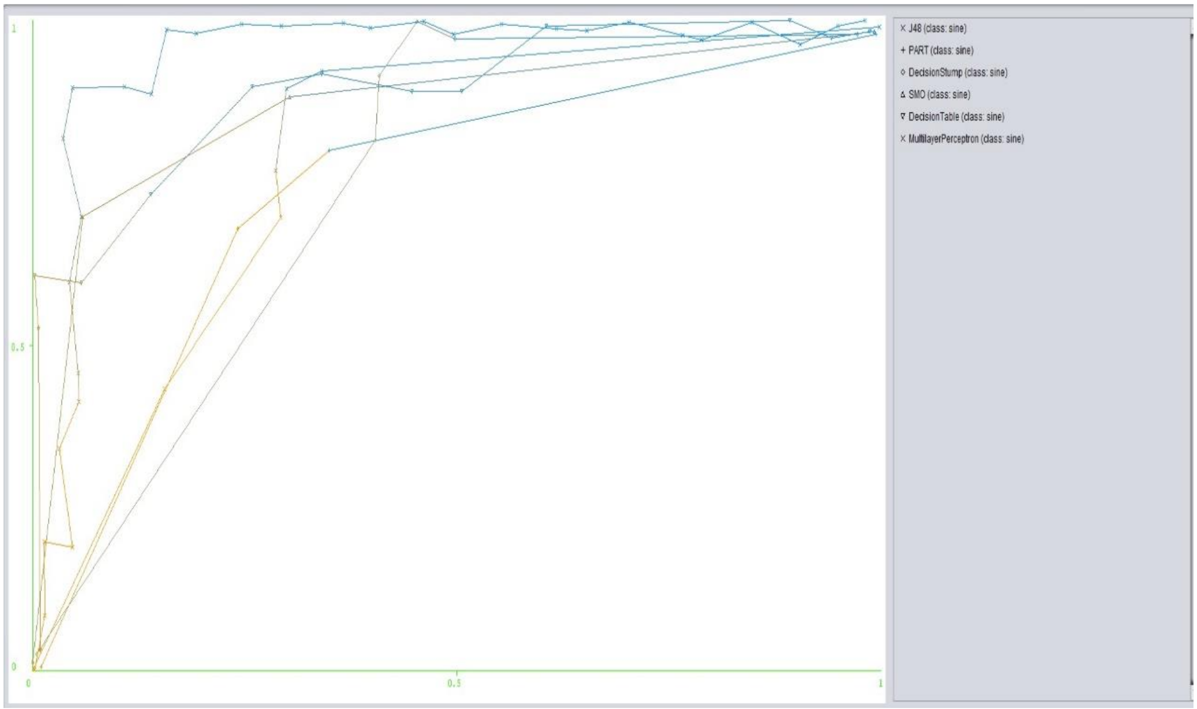


Figure 20: ROC Curve of UnknownSource-Destination-Test2 for class Sine [6]

References

[1] Z. Zhang and R. Zhang, Multimedia Data Mining: A Systematic Introduction to Concepts and Theory, Boca Raton: Chapman & Hall/CRC, 2008 .

[2] W. Hsu, M. L. Lee and J. Zhang, "Image Mining: Trends and Developments," *Journal of Intelligent Information Systems,* vol. 19, no. 1, p. 7–23, 2002.

[3] W. Hu, T. Tan, L. Wang and S. Maybank, "A Survey on Visual Surveillance of Object Motion and Behaviors," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* vol. 34, no. 3, pp. 334 - 352, 2004.

[4] R. A. C. Jimenez, Event Detection in Surveillance Video, Boca Raton: Florida Atlantic University, 2010.

[5] GEUTEBRÜCK , "G-Tect G-Tect/VMX," [Online]. Available: https://www.geutebrueck.com/en_US/g-tect-vmx-31715.html. [Accessed 2 May 2018].

[6] "DSMP Data set Repository," [Online]. Available: http://dsmp.ryerson.ca/datasets/obj_track_data/. [Accessed 2 May 2018].

[7] "Ubuntu," Canonical Ltd, 2018. [Online]. Available: https://www.ubuntu.com/. [Accessed 15 April 2018].

[8] "Ubuntu for desktops," https://www.ubuntu.com/desktop, [Online]. Available: https://www.ubuntu.com/desktop. [Accessed 11 April 2018].

[9] M. Helmke, E. K. Joseph and J. A. Rey, The Official Ubuntu Book, New Jersey: Prentice Hall, 2016.

[10] K. Thomas, Beginning Ubuntu Linux,From Novice to Professional, New York: Apress, 2006.

[11] J. F. Kelly, Ubuntu on a Dime: The Path to Low-Cost Computing, New York: Apress, 2009.

[12] Ubuntu Documentation Project, Ubuntu 10.04 Lts Installation Guide, Palo Alto: Fultus Corporation, 2010.

[13] L. L. Fernández, M. P. Díaz and M. B. Raúl , "Kurento: a media server technology for convergent WWW/mobile real-time multimedia communications supporting WebRTC," in *2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, Madrid, 2013.

[14] "Kurento," kurento.org, [Online]. Available: https://doc-kurento-repository.readthedocs.io/en/latest/#. [Accessed 11 August 2017].

[15] C. R. García, P. C. Gil, M. Burmester and A. Q. Arencibia, Ubiquitous Computing and Ambient Intelligence, Cham: Springer International Publishing, 2016.

[16] "Kurento Media Server," Kurento.org, [Online]. Available: https://www.kurento.org/. [Accessed 11 August 2017].

[17] L. L. Fernández, M. Gallego and B. García, "Authentication, Authorization, and Accounting in WebRTC PaaS Infrastructures: The Case of Kurento," *IEEE Internet Computing*, vol. 18, no. 6, pp. 34 - 40, 2014.

[18] "The LIVE555TM Media Server," Live Networks, Inc, [Online]. Available: http://www.live555.com/mediaServer/. [Accessed 11 August 2017].

[19] C. Wei and H. Zhang, "Applications of a Streaming Video Server in a Mobile Phone Live Streaming System," *Journal of Software Engineering and Applications*, vol. 7, no. 12, pp. 975-982, 2014.

[20] L. Changling, B. Jie and W. Ningguo , "Design and Implementation of Web Video Surveillance System Based on Live555," *Modern Science & Technology of Telecommunications*, vol. 12, no. 1, pp. 30-35, 2012.

[21] G. Bradski and A. Kaehler, Learning OpenCV, vol. 1, Sebastopol: O'Reilly Media, Inc., 2008, pp. 45-50.

[22] S. Brahmbhatt, Practical OpenCV (Technology in Action), New York: Apress, 2013.

[23] A. Kaehler and G. Bradski, Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library, New York: O'Reilly Media, Inc., 2017.

[24] "OpenCV," OpenCV team, [Online]. Available: https://opencv.org/about.html. [Accessed 21 April 2018].

[25] J. Dalal and S. Patel, Instant OpenCV Starter, Birmingham: Packt Publishing Ltd, 2013.

[26] B. Burd, Beginning Programming with Java For Dummies, Hoboken: John Wiley and Sons, Inc, 2012.

[27] D. Flanagan, Java in a Nutshell: A Desktop Quick Reference for Java Programmers, Sebastopol: O'Reilly Media, 1997.

[28] M. Sikora, Java: Practical Guide for Programmers, Burlington: Morgan Kaufmann, 2003.

[29] C. S. Horstmann and G. Cornell, Core Java, Volume II--Advanced Features, London: Pearson Education, 2008.

[30] S. Holzner, PHP: The Complete Reference, New York: McGraw Hill Professional, 2007.

[31] V. Vaswani , PHP: A Beginner's guide, New York: McGraw-Hill Education, 2008.

[32] M. Doyle, Beginning PHP 5.3, New Jersey: John Wiley & Sons, 2009.

[33] E. Quigley, JavaScript by Example, New Jersey: Prentice Hall Professional Technical Reference , 2004.

[34] J. Resig, R. Ferguson and J. Paxton, Pro JavaScript Techniques : Second Edition, New York: Apress, 2014.

[35] K. Sun and S. Ryu, "Analysis of JavaScript Programs: Challenges and Research Trends," *ACM Computing Surveys (CSUR)*, vol. 50, no. 4, 2017.

[36] D. Flanagan, JavaScript: The Definitive Guide, Sebastopol: O'Reilly, 2002.

[37] K. R. Alik, "An Efficient k-Means Clustering Algorithm," *Pattern Recognition Letters,* vol. 29, no. 9, pp. 1385-1391, 2008.

[38] G. Budura, C. Botoca and N. Micl˘au, "Competitive Learning Algorithms for Data Clustering," *Facta universitatis - series: Electronics and Energetics,* vol. 19, no. 2, pp. 261-269, 2006.

[39] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, Berkeley, 1967.

[40] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-Means Clustering Algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics),* vol. 28, no. 1, pp. 100-108, 1979.

[41] "A Tutorial on Clustering Algorithms," [Online]. Available: https://home.deib.polimi.it/matteucc/Clustering/tutorial_html/index.html. [Accessed 11 August 2017].

[42] A. Trevino , "Introduction to K-means Clustering," DataScience, [Online]. Available: https://www.datascience.com/blog/introduction-to-k-means-clustering-algorithm-learn-data-science-tutorials. [Accessed 16 August 2017].

[43] C. Piech, "K Means," Stanford, [Online]. Available: http://stanford.edu/~cpiech/cs221/handouts/kmeans.html. [Accessed 16 August 2017].

[44] R. Xu and D. Wunsch, Clustering, New Jersey: John Wiley & Sons, 2009.

[45] J. M. Pena, J. A. Lozano and P. Larranaga, "An empirical comparison of four initialization methods for the K-Means algorithm," *Pattern Recognition Letters,* vol. 20, no. 10, pp. 1027-1040, 1999.

[46] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman and A. Y. Wu, "An Efficient k-Means Clustering Algorithm: Analysis and Implementation," *IEEE*

Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 881 - 892, 2002.

[47] K. L. Du, "Clustering: A neural network approach," *Neural Networks,* vol. 23, no. 1, pp. 89-107, 2010.

[48] R. Tibshirani, G. Walther and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Statistical Methodology (Series B),* vol. 63, no. 2, pp. 411-423, 2001.

[49] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters,* vol. 31, no. 8, pp. 651-666, 2010.

[50] M. Meilă, "Comparing Clusterings by the Variation of Information," *Learning Theory and Kernel Machines,* vol. 2777, no. 1, pp. 173-187, 2003.

[51] C. J. Burges, "Simplified support vector decision rules," in *International Conference on Machine Learning*, Bari, 1996 .

[52] E. Osuna, R. Freund and F. Girosi, "Support Vector Machines: Training and Applications," *Technical Report,* 1997.

[53] C. Campbell and Y. Ying, Learning with Support Vector Machines, San Rafael: Morgan & Claypool Publishers, 2010.

[54] C. W. Hsu and C. J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks,* vol. 13, no. 2, pp. 415 - 425, 2002 .

[55] C. Stoean and R. Stoean, Support Vector Machines and Evolutionary Algorithms for Classification: Single or Together?, New York: Springer, 2014.

[56] B. H. Boyle, Support Vector Machines: Data Analysis, Machine Learning and Applications, New York: Nova Science Publishers, Inc, 2011.

[57] G. Kaur and A. Chhabra, "Improved J48 Classification Algorithm for the Prediction of Diabetes," *International Journal of Computer Applications,* vol. 98 , no. 22, pp. 13-17, 2014.

[58] H. Seif, "Naïve Bayes and J48 Classification Algorithms on Swahili Tweets: Perfomance Evaluation," *(IJCSIS) International Journal of Computer Science and Information Security,* vol. 14, no. 1, pp. 1-4, 2016.

[59] M. H. Dunham , Data Mining: Introductory and Advanced Topics, New Jersey: Prentice Hall/Pearson Education, 2002.

[60] D. Juneja, S. Sharma, A. Jain and S. Sharma, "A novel approach to construct decision tree using quick C4.5 algorithm," *Oriental Journal of Computer Science & Technology,* vol. 3, no. 2, pp. 305-310, 2010.

[61] N. Bhargava, G. Sharma, R. Bhargava and M. Mathuria, "Decision Tree Analysis on J48 Algorithm for Data Mining," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 3, no. 6, pp. 1114-1119, 2013.

[62] F. Thabtah and P. Cowling, "Mining the data from a hyperheuristic approach using associative classification," *Expert Systems with Applications,* vol. 34, no. 2, p. 1093–1101, 2008.

[63] E. Frank and I. H. Witten , "Generating Accurate Rule Sets Without Global Optimization," in *ICML '98 Proceedings of the Fifteenth International Conference on Machine Learning*, San Francisco, 1998.

[64] G. Pagallo and D. Haussler, "Boolean Feature Discovery in Empirical Learning," *Machine Learning,* vol. 5, no. 1, p. 71–99, 1990.

[65] W. N. Haizan, W. Mohamed, M. N. Mohd Salleh, and A. H. Omar, "A Comparative Study of Reduced Error Pruning Method in Decision Tree Algorithms," in *IEEE International Conference on Control System, Computing and Engineering,*, Penang, 2012.

[66] C. Tan, H. Chen and C. Xia, "The Prediction of Cardiovascular Disease Based on Trace Element Contents in Hair and a Classifier of Boosting Decision Stumps," *Biological Trace Element Research*, vol. 129, no. 1-3, p. 9–19, 2009.

[67] R. Madhu and R. Senthilkumar, "Recommendation System to Accomplish User Pursuit," in *International Conference on Recent Trends in Information Technology*, Chennai, 2014.

[68] V. V. Vijayanrockz and A. Chandrika, "Prediction and Diagnosis of Diabetes Mellitus -A Machine Learning Approach," in *IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, Trivandrum, 2015.

[69] M. Sugiyama , Introduction to Statistical Machine Learning, Waltham: Elsevier Inc., 2015.

[70] L. Rokach, Pattern Classification Using Ensemble Methods, New Jersey: World Scientific Publishing Co., 2010.

[71] D. P. Acharjya and A. Mitra, Bio-Inspired Computing for Information Retrieval Applications, Hershey: IGI Global, 2017.

[72] R. Kohavi, "The power of decision tables," in *European Conference on Machine Learning*, Heraklion, 1995.

[73] G. Wets, J. Vanthienen and H. Timmermans, "Modelling decision tables from data," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, Melbourne, 1998.

[74] R. Hewett and J. Leuchner, "The Power of Second-Order Decision Tables," in *Proceedings of the 2002 SIAM International Conference on Data Mining*, Crystal City, 2002.

[75] G. Wets, J. Vanthienen and S. Piramuthu, "Extending a tabular knowledge-based framework with feature selection," *Expert Systems with Applications*, vol. 13, no. 2, pp. 109-119, 1997.

[76] B. Baesens, R. Setiono, C. Mues and J. Vanthienen, "Using Neural Network Rule Extraction and Decision Tables for Credit-Risk Evaluation," *Management Science,*, vol. 49, no. 3, pp. 312-329, 2003.

[77] M. M. Kirmani, "Heart Disease Prediction using Multilayer Perceptron Algorithm," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 5, 2017.

[78] R. Baele and T. Jackson, Neural Computing: An Introduction, Bristol: IOP Publishing Ltd, 1991.

[79] M. W. Gardner and S. R. Dorling, "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences," *Atmospheric Environment,* vol. 32, no. 14-15, pp. 2627-2636, 1998.

[80] A. T. Schroeder, "Data mining with neural networks: Solving business problems from application development to decision support," *Journal of the American Society for Information Science,* vol. 48, no. 9, pp. 862-863, 1997.

[81] O. Adwan, N. Ghatasheh and H. Faris, "Predicting Customer Churn in Telecom Industry using Multilayer Perceptron Neural Networks: Modeling and Analysis," *Life Science Journal,* vol. 11, no. 3, pp. 75-81, 2014.

[82] K. Kompoliti and L. V. Metman, Encyclopedia of Movement Disorders, Chicago: Elsevier Academic Press, 2010.

[83] R. R. Bouckaert, E. Frank, M. Hall, R. Kirkby, P. Reutemann, A. Seewald and D. Scuse, "WEKA Manual for Version 3-8-1," University of Waikato,, Hamilton, 2016.

[84] C. D. Manning, P. Raghavan and . H. Schütze, Introduction to Information Retrieval, Cambridge: Cambridge University Press, 2008.

[85] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in *ICML '06 Proceedings of the 23rd international conference on Machine learning*, Pittsburgh, 2006.

[86] J. M. Santos and M. Embrechts, "On the Use of the Adjusted Rand Index as a Metric for Evaluating Supervised Classification," in *International Conference on Artificial Neural Networks,* Sofia, 2009.

[87] W. M. Rand, "Objective Criteria for the Evaluation of Clustering Methods," *Journal of the American Statistical Association,* vol. 66, no. 336, pp. 846-850, 1971.

[88] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters,* vol. 27, no. 8, pp. 861-874, 2006.

[89] P. Kumar and A. Tiwari, Ubiquitous Machine Learning and Its Applications, Hershey: IGI Global, 2017.

[90] M. Zahedi and A. Ghanbari Sorkhi, "Improving Text Classification Performance Using PCA and Recall-Precision Criteria," *Arabian Journal for Science and Engineering,* vol. 38, no. 8, p. 2095–2102, 2013.

[91] C. J. V. Rijsbergen, Information Retrieval, Newton: Butterworth-Heinemann, 1979 .

[92] H. Xiong, J. Wu and J. Chen, "K-Means Clustering Versus Validation Measures: A Data-Distribution Perspective," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics),* vol. 39, no. 2, pp. 318 - 331, 2009.

[93] W. J. Krzanowsk and D. J. Hand, ROC Curves for Continuous Data, Boca Raton: CRC Press, 2009.

[94] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognition,* vol. 30, no. 7, pp. 1145-1159, 1997.

[95] J. Davis and M. H. Goadrich, "The relationship between Precision-Recall and ROC curves," in *ICML 06 Proceedings of the 23rd international conference on Machine learning,* Pittsburgh, 2006.

[96] D. Chu, C. h. Jiang, Z. B. Hao and W. Jiang, "The Design and Implementation of Video Surveillance System Based on H.264, SIP,RTP/RTCP and RTSP," in *2013 Sixth International Symposium on Computational Intelligence and Design,* Hangzhou, 2013.

[97] H. Schulzrinne, A. Rao and R. Lanphier, "Real Time Streaming Protocol (RTSP)," The Internet Society, Reston, 1998.

[98] B. Hdioud, A. Ezzahout and Y. Hadi, "A real-time people tracking system based on trajectory estimation using single field of camera view," in *2013 International Conference on Computer Applications Technology (ICCAT)*, Sousse, 2013.

[99] W. Niu, J. Long, D. Han and Y. F. Wang, "Human Activity Detection and Recognition for Video Surveillance," in *IEEE International Conference on Multimedia and Expo (ICME),* Taipei, 2004.

[100] S. Zhang, C. Wang and S. C. Chan, "New Object Detection, Tracking, and Recognition Approaches for Video Surveillance Over Camera Network," *IEEE SENSORS JOURNAL,*, vol. 15, no. 5, pp. 2679 - 2691, 2015.

[101] R. R. Sillito and R. B. Fisher, "Semi-supervised Learning for Anomalous Trajectory Detection," in *Proceedings British Machine Vision Conference BMVC2008*, Leeds, 2008.

[102] C. Piciarelli, C. Micheloni, and G. L. Foresti, "Trajectory-Based Anomalous Event Detection," *IEEE Transactions on Circuits and Systems for Video Technology* , vol. 18, no. 11, pp. 1544 - 1554, 2008.

[103] A. Naftel and S. Khalid, "Classifying spatiotemporal object trajectories using unsupervised learning in the coefficient feature space," *Multimedia Systems,* vol. 12, no. 3, p. 227–238, 2006.

[104] L. A. Elrefaei, M. O. Al-musawa and N. A. Al-gohany, "Development of An Android Application for Object Detection Based on Color, Shape, or Local Features," *The International Journal of Multimedia & Its Applications,* vol. 9, no. 1, pp. 21-30, 2017.

[105] G. Medioni, I. Cohen, F. Bremond , S. Hongeng and R. Nevatia, "Event detection and analysis from video streams," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 23, no. 8, pp. 873 - 889, 2001.

[106] C. Piciarelli and G. L. Foresti, "On-line trajectory clustering for anomalous events detection," *Pattern Recognition Letters,* vol. 27, no. 15, p. 1835–1842, 2006.

[107] P. K. Mishra and G. P. Saroha, "A study on video surveillance system for object detection and tracking," in *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, 2016.

[108] I. Haritaoglu, D. Harwood and L. S. Davis, "W4: Who? When? Where? What? A Real Time System for Detecting and Tracking People," in *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, Nara, 1998.

[109] N. Johnson and D. Hogg, "Learning the distribution of object trajectories for event recognition," *Image and Vision Computing,* vol. 14, no. 8, pp. 609-615, 1996.

[110] A. Rosebrock, "Ball Tracking with OpenCV," pyimageresearch, 14 09 2014. [Online]. Available: https://www.pyimagesearch.com/2015/09/14/ball-tracking-with-opencv/. [Accessed 15 03 2018].

[111] A. Abhari, S. Banihashemi and J. Li, "Object Movement Detection by Real-Time Deep Learning for Security Surveillance Camera," in *IEEE Computer Soceity, 2017 International Conference on Computational Science and Computational Intelligence,* Las Vegas, 2017.

[112] I. H. Witten, E. Frank, L. Trigg, M. Hall, G. Holmes and S. J. Cunningham, "Weka: Practical Machine Learning Tools and Techniques with Java Implementations," The University of Waikato, Hamilton, 1999.

[113] G. Kohlberg and M. Hammer, "GetTheDiagnosis.org: A Database of Sensitivity and Specificity," [Online]. Available: http://getthediagnosis.org/roc.html. [Accessed 15 1 2018].

[114] "TRENDnet," TRENDnet, Inc, [Online]. Available: https://www.trendnet.com/products/ip-cameras/TV-IP672WI. [Accessed 30 April 2018].

[115] "Overview-Weka," [Online]. Available: http://weka.sourceforge.net/doc.stable/. [Accessed 10 1 2018].

[116] "OpenCV 3.1.0, Open Source Computer Vision," doxygen, [Online]. Available: http://docs.opencv.org/3.1.0/. [Accessed 11 August 2017].

[117] "How To Install Linux, Apache, MySQL, PHP (LAMP) stack on Ubuntu 16.04," DigitalOcean, [Online]. Available: https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-16-04. [Accessed 11 August 2017].