Automatic Video Annotation Tool

Via Supervised Learning

By

Ali Yar Khan

Supervised By

Dr. Umer Rashid

Department of Computer Science,

Quaid-i-Azam University, Islamabad, Pakistan.

Session (2015-2019)

Table of Contents

List of tables

List of figures

CHAPTER 1: SOFTWARE PROJECT MANAGEMENT PLAN

1. Software Project Management Plan

1.1 Introduction:

A software program management proposal could be a management legal paper for managing software projects; It explains the software methods, milestones, and different essential details for the progress of software work results that meet the product requirements.

1.1 Project Overview:

Automated video annotation tool is a desktop application based on classifier that automate the process of annotating the videos based on the concepts (contents) of the video.

1.2 Project Deliverables:

The project deliverables consist of the following documents:

1. Software Project Management Plan
2. Software Requirements Specifications
3. Software Design Description
4. Software Test Documentation
5. Realize

1.3 Problem Statement:

Today, data is used to make access easier (i.e. by tagging). This is one of the key ideas behind the knowledge network. The data contained in the metadata allows for more efficient searches. As far as video is concerned, statistics are becoming a huge day of the day, and if fully explained by text description, video search can be done more efficiently than text search.

1.5 Purpose of product:

Automated Video Annotation tool will facilitate its users by automatically annotating the videos given to it. This will not only reduce the workload but will also make retrieval of videos efficient.

1.6 Definitions, Acronyms and Abbreviations:

Table 1 Definition, Acronyms and Abbreviations

Term	Definition
Annotation	The keywords associated to videos as metadata.
Matlab	It is a product of math works. It has a extreme level language and cooperating environment that enables you to achieve the tasks you perform.
SA-ITVA	Semi-Automatic Interactive tool for video annotation.
Classification Modal	Used for Support Vector Machine (SVM).
IEEE	Institute of Electrical and Electronics Engineers.
UI	User Interface.

1.7 Objectives:

The major goals of this project are:

- Development of an interactive tool which provides users with a simple and convenient interface for annotation.
- Tool will use multiple features sets of videos like visual, audio (speech and music) as input of annotation.
- Allow user to create, load and save the Classification Modal.
- Tool will annotate the video using Artificial Intelligence concept known as Classification Modal.
- Allow user to give his feedback on annotations suggested by classifier.
- Allow user to judge and choose the relevant keywords, to correct them, to revise them, and to add some other annotations.
- Tool will learn and adapt from the user suggestions and improve the performance in the next annotations.

1.8 Scope:

An interactive tool for Automated Video Annotation (IT-AVA) will allow its user to select a video and see its annotations. The tool will extract key frames from which it extracts visual low level features or descriptors of these key frames. These features are mapped to the defined concepts using already trained classifier. This classifier is trained on a set of images with their annotations. Classifier will suggest concepts related to the given video using its training on the training dataset.

1.8.1 Inputs:

- Video with labels (for learning).
- Video which is to be annotated.

1.8.2 Output:

- Labels for given video.

1.8.3 Functionalities:

- Learn the concepts from the video and labels given to it.
- Label the video given to it.

1.9 Project Organization:

Project Organization contains description of software process model used for the project, roles the people play in the making of the project and tackles and methods to be used in this project.

1.9.1 Software Process Model:

For this project, Prototyping process model will be used. Reason for using it is that requirements are not clear and more functionalities can be added subsequently.

1.10 Roles and Responsibilities:

Development of this product is my responsibility whereas tools and techniques to develop this product is provided by the supervisor.

1.11 Tools and Techniques:

Following tools and techniques are employed in the project

- MS Word 2013.
- ProjectLibre 1.6.2.

- Matlab.

1.12 PROJECT MANAGEMENT PLAN:

Project Management strategy explains the task, the deliverables and milestones and resources needed to complete the task.

1.12.1 Tasks:

Following is the list of tasks for the project:

- **Problem Understanding**
    - **Description:**

      First problem definition is must.
    - **Deliverables and Milestones:**

      None.
    - **Resources Needed:**

      People: Ali Yar Khan, Supervisor.
    - **Dependencies and Constraints:**

      None.
- **Software Project Management Plan**
    - **Description:**

      Secondly software methodology and objectives are identified.
    - **Resources Needed:**

      People: Ali Yar Khan, Supervisor.

      Software: MS Word, Project Libre.
    - **Dependencies and Constraints:**

      Problem Understanding.
    - **Deliverables and Milestone**

      Project Management Plan.
- **Software Requirement Specification**
    - **Description:**

      Thirdly analysis on how the requirements will meet is included.

- **Deliverables and Milestones:**

  SRS Document.

- **Resources Needed:**

  People: Ali Yar Khan, Supervisor.

  Software: MS Word

- **Dependencies and Constraints:**

  Software Project Management Plan

- **Risks and Contingencies:**

  None.

- **Software Design Description**
  - **Description:**

    Fourthly detailed design and interface design will be included.

  - **Resources Needed:**

    People: Ali Yar Khan, Supervisor.

    Software: MS Word, Argo UML.

  - **Dependencies and Constraints:**

    Analysis and Requirement.

  - **Deliverables and Milestone:**

    Software Design Document.

- **Software Test Documentation**
  - **Description:**

    In this part test plans and test case to prove and confirm the tool and their results will be included.

  - **Deliverables and Milestones:**

    Software Design Description and Software Test Documentation.

  - **Resources Needed:**

    Ali Yar Khan.

- **Dependencies and Constraints:**

  Software Design Description.

- **Software Implementation**
  - **Description:**

    How the system will be implemented.

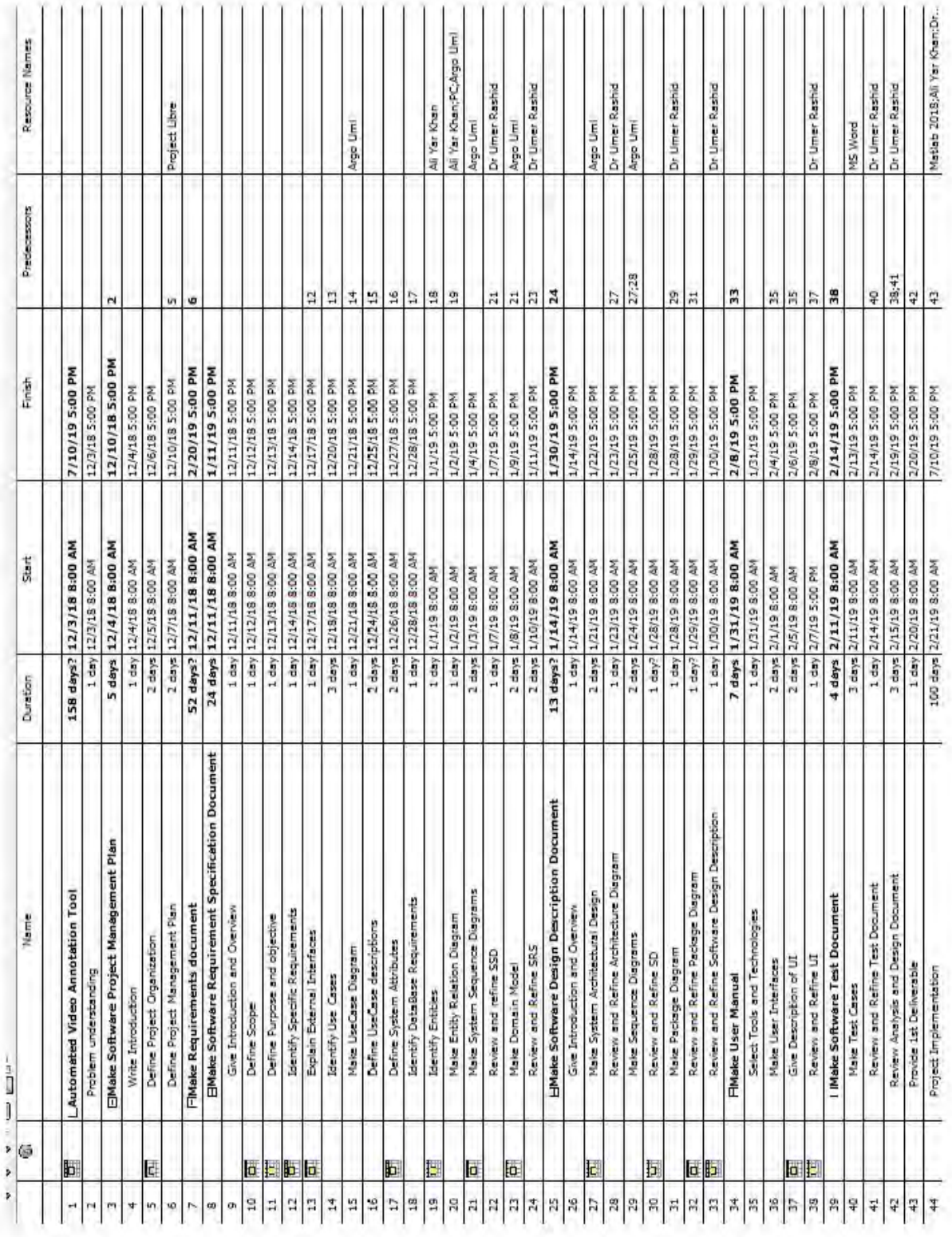
  - **Resources Needed:**

    Ali Yar Khan

  - **Dependencies and Constraints:**

    Software Test Documentation

1.13 Timetable and Gantt chart:



	Name	Duration	Start	Finish	Predecessors	Resource Names
1	Automated Video Annotation Tool	158 days?	12/3/18 8:00 AM	7/10/19 5:00 PM		
2	Problem understanding	1 day	12/3/18 8:00 AM	12/3/18 5:00 PM		
3	Make Software Project Management Plan	5 days	12/4/18 8:00 AM	12/10/18 5:00 PM	2	
4	Write Introduction	1 day	12/4/18 8:00 AM	12/4/18 5:00 PM		
5	Define Project Organization	2 days	12/5/18 8:00 AM	12/6/18 5:00 PM		
6	Define Project Management Plan	2 days	12/7/18 8:00 AM	12/10/18 5:00 PM	5	Project Libre
7	Make Requirements document	52 days?	12/11/18 8:00 AM	2/20/19 5:00 PM	6	
8	Make Software Requirement Specification Document	24 days	12/11/18 8:00 AM	1/11/19 5:00 PM		
9	Give Introduction and Overview	1 day	12/11/18 8:00 AM	12/11/18 5:00 PM		
10	Define Scope	1 day	12/12/18 8:00 AM	12/12/18 5:00 PM		
11	Define Purpose and objective	1 day	12/13/18 8:00 AM	12/13/18 5:00 PM		
12	Identify Specific Requirements	1 day	12/14/18 8:00 AM	12/14/18 5:00 PM		
13	Explain External Interfaces	1 day	12/17/18 8:00 AM	12/17/18 5:00 PM	12	
14	Identify Use Cases	3 days	12/18/18 8:00 AM	12/20/18 5:00 PM	13	
15	Make UseCase Diagram	1 day	12/21/18 8:00 AM	12/21/18 5:00 PM	14	Argo Uml
16	Define UseCase descriptions	2 days	12/24/18 8:00 AM	12/25/18 5:00 PM	15	
17	Define System Attributes	2 days	12/26/18 8:00 AM	12/27/18 5:00 PM	16	
18	Identify DataBase Requirements	1 day	12/28/18 8:00 AM	12/28/18 5:00 PM	17	
19	Identify Entities	1 day	1/1/19 8:00 AM	1/1/19 5:00 PM	18	Ali Yar Khan
20	Make Entity Relation Diagram	1 day	1/2/19 8:00 AM	1/2/19 5:00 PM	19	Ali Yar Khan;PC;Argo Uml
21	Make System Sequence Diagrams	2 days	1/3/19 8:00 AM	1/4/19 5:00 PM		Argo Uml
22	Review and refine SSD	1 day	1/7/19 8:00 AM	1/7/19 5:00 PM	21	Dr Umer Rashid
23	Make Domain Model	2 days	1/8/19 8:00 AM	1/9/19 5:00 PM	21	Argo Uml
24	Review and Refine SRS	2 days	1/10/19 8:00 AM	1/11/19 5:00 PM	23	Dr Umer Rashid
25	Make Software Design Description Document	13 days?	1/14/19 8:00 AM	1/30/19 5:00 PM	24	
26	Give Introduction and Overview	1 day	1/14/19 8:00 AM	1/14/19 5:00 PM		
27	Make System Architectural Design	2 days	1/21/19 8:00 AM	1/22/19 5:00 PM		Argo Uml
28	Review and Refine Architecture Diagram	1 day	1/23/19 8:00 AM	1/23/19 5:00 PM	27	Dr Umer Rashid
29	Make Sequence Diagrams	2 days	1/24/19 8:00 AM	1/25/19 5:00 PM	27;28	Argo Uml
30	Review and Refine SD	1 day?	1/28/19 8:00 AM	1/28/19 5:00 PM		
31	Make Package Diagram	1 day	1/28/19 8:00 AM	1/28/19 5:00 PM	29	Dr Umer Rashid
32	Review and Refine Package Diagram	1 day?	1/29/19 8:00 AM	1/29/19 5:00 PM	31	
33	Review and Refine Software Design Description	1 day	1/30/19 8:00 AM	1/30/19 5:00 PM		Dr Umer Rashid
34	Make User Manual	7 days	1/31/19 8:00 AM	2/8/19 5:00 PM	33	
35	Select Tools and Technologies	1 day	1/31/19 8:00 AM	1/31/19 5:00 PM		
36	Make User Interfaces	2 days	2/1/19 8:00 AM	2/4/19 5:00 PM	35	
37	Give Description of UI	2 days	2/5/19 8:00 AM	2/6/19 5:00 PM	35	
38	Review and Refine UI	1 day	2/7/19 5:00 PM	2/8/19 5:00 PM	37	Dr Umer Rashid
39	Make Software Test Document	4 days	2/11/19 8:00 AM	2/14/19 5:00 PM	38	
40	Make Test Cases	3 days	2/11/19 8:00 AM	2/13/19 5:00 PM		MS Word
41	Review and Refine Test Document	1 day	2/14/19 8:00 AM	2/14/19 5:00 PM	40	Dr Umer Rashid
42	Review Analysis and Design Document	3 days	2/15/19 8:00 AM	2/19/19 5:00 PM	38;41	Dr Umer Rashid
43	Provide 1st Deliverable	1 day	2/20/19 8:00 AM	2/20/19 5:00 PM	42	
44	Project Implementation	100 days	2/21/19 8:00 AM	7/10/19 5:00 PM	43	Matlab 2018;Ali Yar Khan;Dr...

Figure 1 Timetable

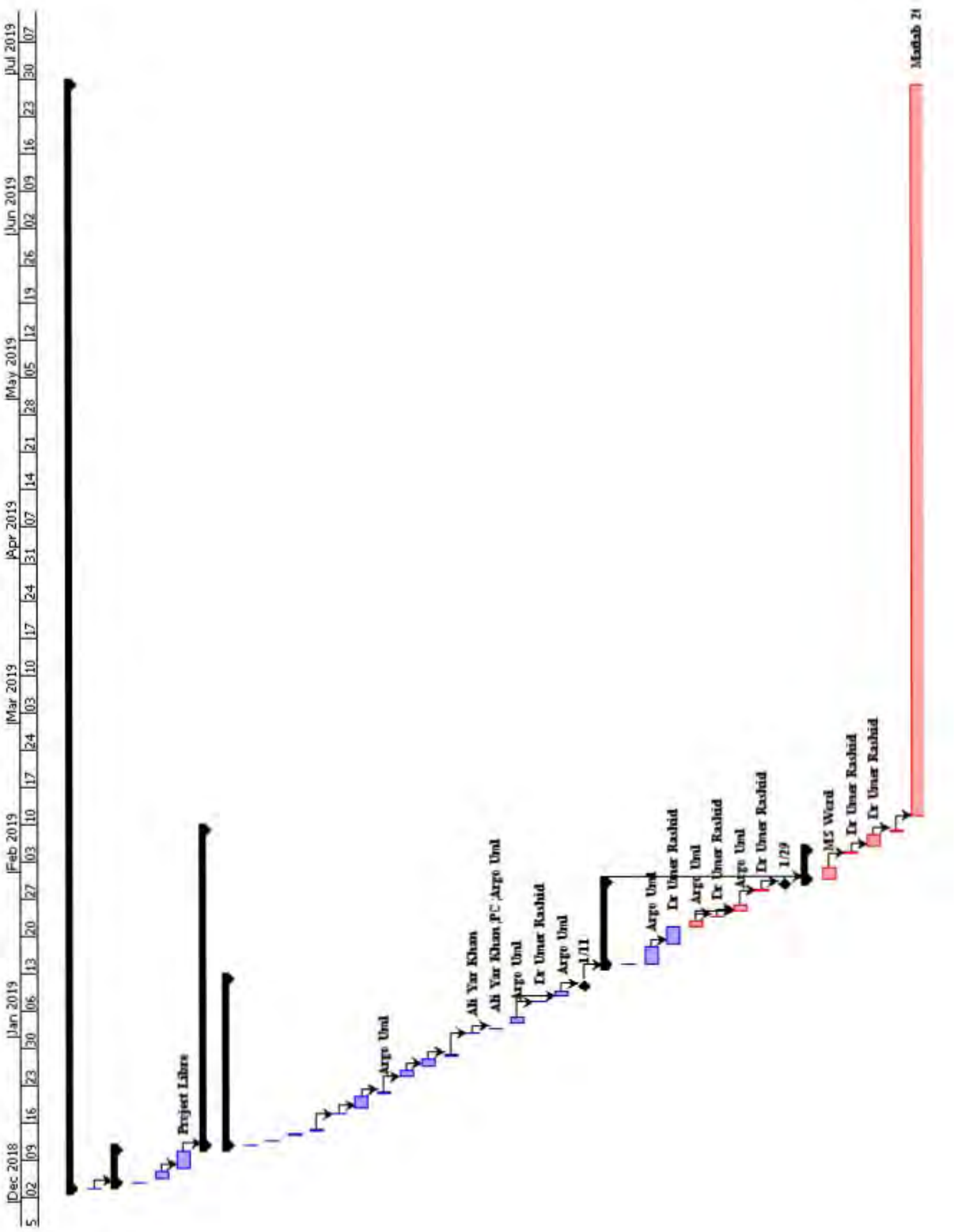


Figure 2 Gantt Chart

CHAPTER 2: SOFTWARE REQUIREMENT SPECIFICATION

2 Software Requirement Specification

2.1 Introduction:

Define the needs of the software system to develop the software requirements (SRS). It removes active and inactive supplies and may contain a set of usage cases that will be provided that explain the user's conversation al-Bay Regi software.

2.2 Product overview:

Automated video annotation tool (AVAT) is implemented in Matlab version R2018a. This product will annotate the videos based on the contents of the video.

2.3 Specific Requirements:

2.3.1 Functional Requirements:

Functional necessities are the product abilities that has got to be offered for the client so as to finish the services provided by the system and mention however the system ought to reply to the error conditions or invalid inputs. The system provides the correct message of any invalid entry. Main purposeful necessities of the system are:

- Training a classifier.
- Retrain a pretrained classifier
- Save a classifier
- Load a classifier
- Create dataset.
- Annotating video
- Save annotations

2.3.2 User Interfaces:

User Interfaces will help the user of the system to interact with the system efficiently. User interface of Automatic Video Annotation Tool (AVAT) are build in MATLAB app designer.

2.3.3 External Interface Requirements:

This section provides an outline of all inputs into the system and outputs from the system. It conjointly offers an outline of the hardware, code and communication.

2.3.3.1 Hardware Requirements:

It requires at least Corei3 with 8 GB of RAM to run smoothly this application.

Note: These are minimum requirements for this application on which it is developed.

2.3.3.2 Software Requirements:

It will be developed for desktop. User with windows 10 and Matlab 2018 (or newer version) can install and use this application.

2.3.3.3 Communication Protocols:

As this application is desktop based, it doesn't communicate over the network whatsoever. So, no communication protocols are needed.

2.4 Use Cases List:

1. Train a classification modal.
2. Retrain a classification modal.
3. Save a classification modal.
4. Load a classification modal.
5. Annotate a video.
6. Save annotation.
7. Update annotations.
8. Add annotations.
9. Create dataset.

2.5 Use Case Diagram:

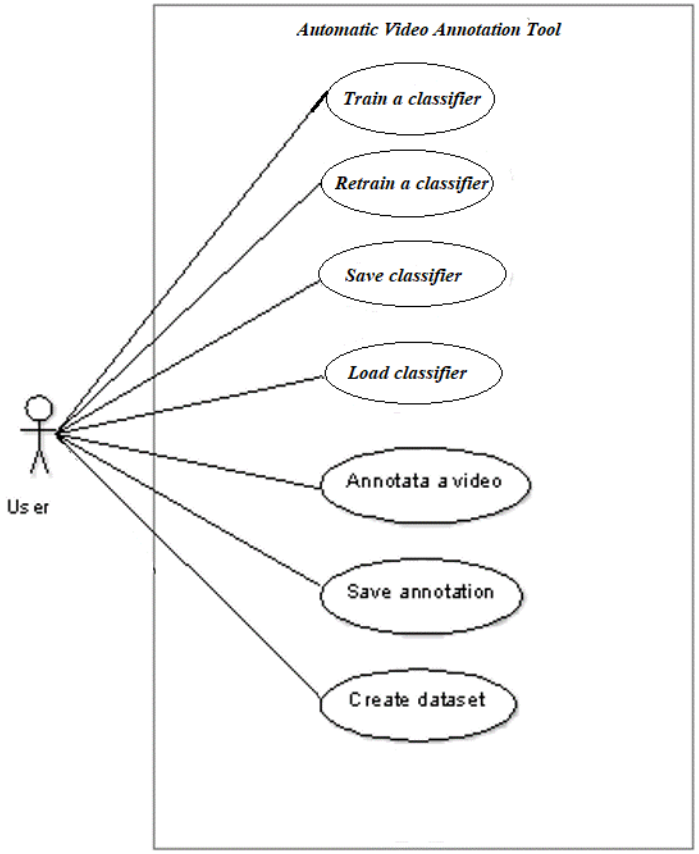


Figure 3 Use Case Diagram

2.6 Use Case Descriptions:

2.6.1 UC-01: Train a Classification Modal:

Table 2 Train a Classification Modal

ID:	UC-01.
Name:	Train a Classification Modal.
Primary actor:	User.
Pre-condition:	Dataset is created/prepared.
Post-condition:	Classification modal is trained.
Input:	Feature vector of images and their annotations.
Output:	Trained classifier.
Destination:	Hard Disk.
Main success scenario:	1. User open “Train” tab. 2. User browses and selects the dataset. 3. User press the “Train” button. 4. System starts the training and prompts the user to wait. 5. Now modal is trained on dataset and saved for later use.

2.6.2 UC-02 Retrain a Classification Modal:

Table 3 Retrain a Classification Modal

ID:	UC-02.
Name:	Retrain the Classification modal.
Primary actor:	User.
Pre-condition:	Dataset is created/available.
Post-condition:	classification modal is trained on the dataset.
Input:	Feature vector of images and their annotations.
Output:	Trained classifier.
Destination:	Hard Disk.
Main success scenario:	1. User open “Train” tab. 2. User browses and selects the dataset. 3. User press the “Train” button. 4. System starts the training and prompts the user to wait. 5. Now modal is trained on dataset and saved for later use.

2.6.3 UC-03 Save a Classification Modal:

Table 4 Save a Classification Modal

ID:	UC-03.
Name:	Save a Classification modal.
Primary actor:	User.
Pre-condition:	Classification modal is already created and trained.
Post-condition:	Classification modal is saved in file on the hard disk.
Input:	File name on which the classifier is to be saved.
Output:	Classification modal is saved onto the hard disk.
Main success scenario:	1. User open “Save modal” option. 2. User browse for location where to save the modal. 3. User gives unique name to the modal. 4. Classification modal is saved on the specified location with the specified name.

2.6.4 UC-04 Load a Classification Modal:

Table 5 Load a Classification Modal

ID:	UC-04.
Name:	Load a Classification Modal.
Primary actor:	User.
Pre-condition:	Trained Classification Modal is already saved.
Post-condition:	Classification Modal is loaded into the system.
Input:	File containing the saved classifier.
Output:	Classification modal is loaded into the system.
Main success scenario:	1. User browses for classification modal file. 2. User selects the file. 3. System prompts the user to wait. 4. Classification modal is loaded into the system.

2.6.5 UC-05 Save Annotations:

Table 6 Save Annotations

ID:	UC-05.
Name:	Save Annotations.
Primary actor:	User.
Pre-condition:	A video is annotated and annotations are displayed.
Post-condition:	Annotations are saved in the file.
Input:	Name of the file and directory where the file is saved.
Output:	Annotations in file.
Destination:	Hard disk.
Main success scenario:	1. User gave a name to the file in which annotations are going to be saved. 2. User press the “Save Annotation” button. 3. User select the directory where to save the file. 4. System saves the annotations in the file.
Extensions:	1a). if user doesn’t give any name to the file, system will prompt “Please provide the file name”.

2.6.6 UC-06 Create Dataset:

Table 7 Create Dataset

ID:	UC-06
Name:	Create dataset.
Primary actor:	User.
Pre-condition:	Different object Images are placed in separate folder.
Post-condition:	System has created a dataset.
Inputs:	Images and their annotations.
Output:	Dataset.
Destination:	Hard disk.
Main success scenario:	1. User open "Create Dataset" option, and chooses directory of images and other where to save the dataset. 2. User selects create dataset option and features of images are extracted. After that and dataset is created along with features and labels.
Extensions:	1a). if user browses but doesn't chooses any directory, then system displays a message, "directory not selected properly". 1b). if user chooses a directory which doesn't have any image, then system will display a message "Directory doesn't have any images".

2.7 Software System Attributes:

This section describes the inactive requirements that are used to estimate the performance of the system.

2.7.1 Reliability:

Since it is based on Classification Modal, its reliability is dependent on the dataset used for its training. Dataset must be good and large for more reliability and better accuracy.

2.7.2 Availability:

The system should be accessible to the user round the clock. No such constraints can affect the availability service of this tool.

2.7.3 Maintainability:

During the development period, all the code will be properly documented so that we can easily make changes and upgrade our application.

2.8 Database Requirements:

Tool will save the features and annotations for training in mat-file and annotations in text file.

Chapter 3: Technical Review

3. Technical Review

3.1 Introduction:

This section describes the existing systems that the user currently can use to comment on video. Highlights its features and limitations here.

3.2 Existing systems:

3.2.1 VATIC :

Inaction is a very much interpretable tool. An old device is not out of the box (due to the unrecognized harmonize dyspepsia) version, but the arrangement can be painful, and you should go into the stoic and java script code you want to keep in the - any problem you have to keep. The video was turned into a series of jpeg pictures, with some staff complaining about the taelong load Tommy.

3.2.2 And a bit:

Like VATIC, you can follow local video and support points and group comments. Very orderly (for example, you can add worldly features), but somehow you cannot provide your label in the comments. There is no way to work between more than one worker.

3.2.3 Scalable:

Promise tools, but hard to use, very wrong. For example, try to comment on something again, but in some frames the creation of dozens of unique identities ends.

3.2.4 Beaver Dam:

A brilliant UCLA device, running as a local Django server, can be merged with mturk. However, it is not clear how to download comments and encountered bugs "as appropriately" from operating (we found and fixed these bugs in a short time, but it can be recommended withcaution)..

3.2.5 YouTube Video Annotation:

Free web-based tool, however you merely annotate YouTube videos that you have uploaded while others can watch them. Text annotation ("text bubbles" or notes), highlight a part of the screen. All annotations can be edited.

3.2.6 LabelMe:

This project, created by Massachusetts Institute of Technology and computer science Laboratory (CSAIL), provides an evidence of a database of digital images. This set of statistics is dynamic, liberated to use and open to the public. Compatible internet browsers and Java scripts need support. In addition, you wish to draw in multiple regions to clarify the objection. [5]

3.2.7 GTVT:

Earth's integrity verification device is used to establish and verify earth's truth data for monitoring systems. It is used as a verification and exactitude mensuration tool for multiple video police work applications [6].

3.2.8 Frame Trail:

Free, but you must host the software. Support for the latest browser as web-based and Fire Fox, Chrome. Safari and opera were not tested. Does not support Internet Explorer [7].

3.3 Currently used techniques:

Currently used techniques [8] for video annotations are:

3.3.1 Seamless annotation & enrichment of mobile captured streams:

These technologies suggest mobile video stream. The system provides users with real-time tag produced automatically from captured videos at this time. It works as a consumer server system in real time. In this way, a central server links the content of the most video frame to the database offered on the web. The possible tag is like capturing the video and sending it back to the user's mobile [9].

3.3.2 Semi-automatic video content annotation:

This technology [10] uses semi-automatic interpretation technology that uses completely different video dispensation ways to search out the visual video background or interpretation scenario. A video view detection algorithm that participates in visual and visual sciences is applied to pay for the results of visualization and interpretation. The videos are shown using five-fold organizational

structures (video, view, group, shots, frames), which express the growing content from the bottom in the granulate Video content is described by such four content descriptors Video Descriptor (Wood) gives full video closure, group descriptor (s) gives the event information, Shot Description (SD) a shot and frames different actions in the descriptor

3.3.3 Automatic video annotation via hierarchal topic trajectory model:

This methodology uses the theme speed model, (11) (heat) to urge a top-ranking model that dynamically changes, that represents the relationship between the video frame and therefore the hooked-up text label. This methodology contains links between visual and text info and 2 the dynamics of the video world. Hits contain four layers 1) statistic numbers, cherish video frames, text labels, and audio signals. 2) knowledge functionality 3) variable and 4) state variables.

As shown in the figure, the HTTM model provides video frames and text labels at the bottom:

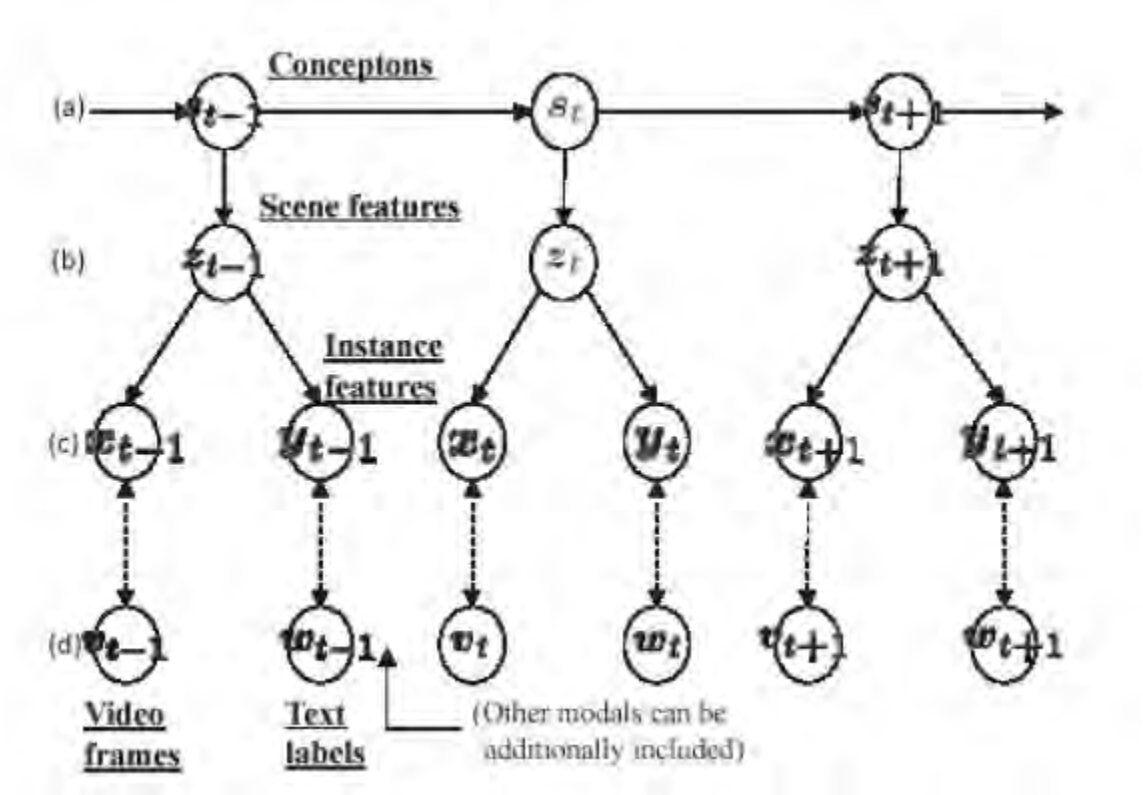


Figure 4 Overview of HTTM model

The features extracted from the video frame and text label are given as X and Y in the next range. The third cover contains hidden variable z, which represents the relationship between visual and audio functions. The last curtain contains hidden mode series which provide variable variables.

3.3.4 Approach to detect text and caption in videos:

This [12] technique uses associate angle-based approach to detect moving captions and text in an exceedingly video. totally different options are wont to describe the realm of text designed by corner. The formula will be wont to detect the transmission of subtitles. Motion functions captured by the sunshine stream then combined with text functions to sight mobile caption patterns. The decision is to grasp the standard of tree classification. Typically, text and titles get better, additional direct data concerning data, time, and location.

3.3.5 Automatic annotation of web services:

In this way, the focus is on the inventory, considering the videos with hard-to-move items and only one objection in the preview. The opening is that the agreement Preview Object Template (CFOT) which incorporates 2 steps 1) preview space estimate and 2) the CFOT algorithmic rule. modification Scale Invariant Feature (SIFT) may be a common algorithm to detect attention-grabbing native points and provides them connected visual details. The Instagram-based gradients (HOG) are designed to get information and visual information from video frames. Audio features are also used to improve accuracy [13].

3.3.6 Ontology-based automatic video annotation technique:

This technique uses ontologies to introduce video collection and sub-processes in smart TV environments. Two types of ontologies are used 1) for knowledge of the domain to be knowledgeable. LSCOM (mass-concept for multimedia knowledge) is used to interpret the content of the knowledge [14] which provides for viewing the quality for video interpretation. The explanation cannot run on every frame, so the first pass is detected by the algorithm [15]. High-level visualization is performed to perform separation, including 1) detecting the inaction frame and objecting: it is employed to calculate the similarity between frames victimization the yukladiaq sig-fish steps. 2) Semi-concept map: Low-level visual options are extracted by MPEG-7 descriptors and their semi-concept values are determined. To remove the color of the features semi-concept map (CSD) [16], the semi-concept mapping is used to map the HHd [17] shape semi-

concept map (SD). 3) Classification rules are used for high-level visualization extraction and definitions.

3.3.7 General structure for video annotation via semi-supervised learning:

This methodology [18] uses semi-instruction learning techniques for attention-grabbing event interpretation in videos. There are 3 functions within the system 1) Event Recognition 2) Event Local3) Search and Navigation of Knowledge. a quick graph-based semi-supervised multiple examples learning (FGSSMIL) rule is providing is together targeted on coaching models to go looking for small-scale expert-labeled videos and large-scale unlabeled videos. FGSMIL addresses the matter of rising detection performance and inadequate coaching information from web resources.

3.3.8 Fast Semantic Diffusion for Large-Scale Context-Based Image & Video annotation:

This methodology [19] uses the construct of data to effectively improve mass pictures and video interpretation. The configuration of the graph spread is used to improve the visualization score. The system has input prediction scores, or manual labels labeled by web users. Knowledge means a weight graph in which it is represented by concepts and weights. This graph is applied to the knowledge retrieval to explain the results of the interpretation. SD is expanded to improve the results of interpretation and is called domain assistant SD (DAS).

Chapter 4 Software Design Description

4. Software Design Description

4.1 Introduction:

A software design document is a document that provides documentation to help develop software and provide details on how to build it. The software design documentation contains software design instructions and graphical documentation that contains details on how to re-complete the software.

4.2 Design Overview:

The software design document provides design details of Semi-Automatic-Interactive Video Explanation Tool. The document contains a complete low-level depiction of the tool, providing insight into the structure and design of each component.

4.3 Design Approach:

Mainly there are two approaches for designing software. It can be design using OOP or structured. OOP design is preferred when systems are large and complex. In building AVAT, we will use structured approach as it is a small and simple system.

4.4 System architecture Design:

A system architecture is an conceptual model that explains the structure, attitude, and the greatest representation of a system. An architecture explain sit supports the structure and logic of behavior of the system, which provides formal explanation and representation of the system.

4.4.1 Architectural Diagram:

The design of the system is shown within the following diagram:

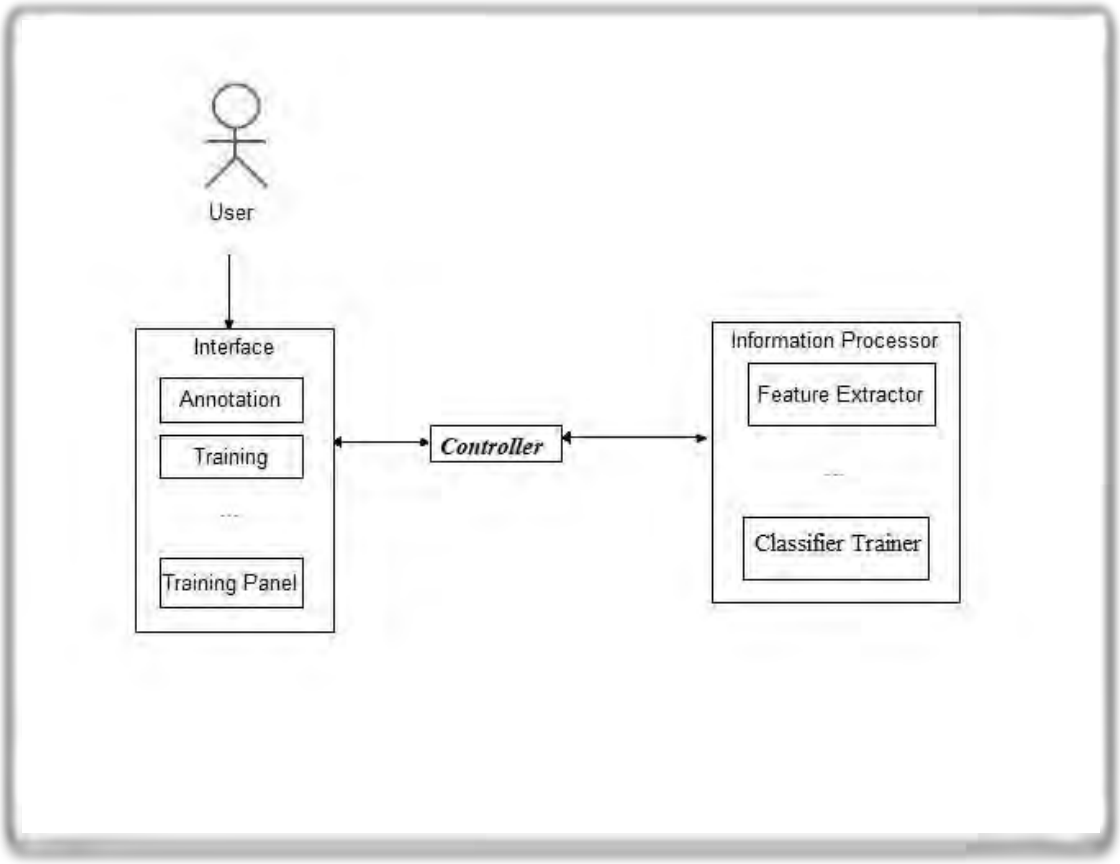


Figure 5 Architecture Diagram

4.5 System Sequence diagrams:

An interface graph which reveals the sequence of dealings among the exterior performer and the system and the events generated by actors is called System Sequence Diagram.

4.5.1 Train a Classification Modal:

Following Figure shows the system sequence diagram of training a classification modal:

Figure 6 SSD for Train a Classification Modal

4.5.2 Retrain a Classification Modal:

Following Figure shows the system sequence diagram of retraining a classification modal:

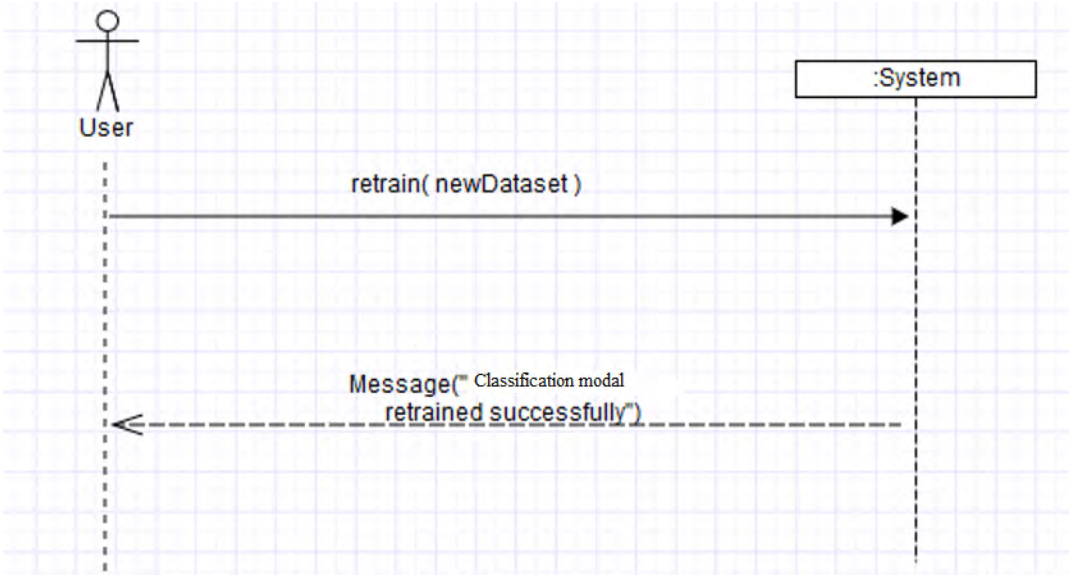


Figure 7 SSD for Retrain a Classification Modal

4.5.3 Save Classification Modal:

Following Figure shows the system sequence diagram of saving a classification modal:

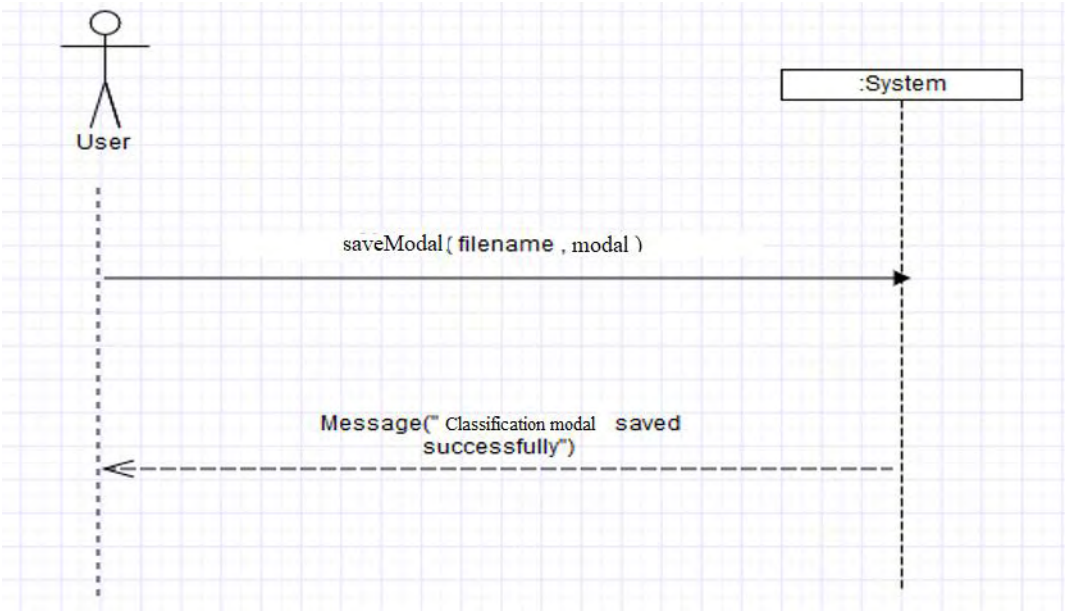


Figure 8 SSD for Save a Classification Modal

4.5.4 Load a Classification Modal:

Following Figure shows the system sequence diagram of loading a classification modal:

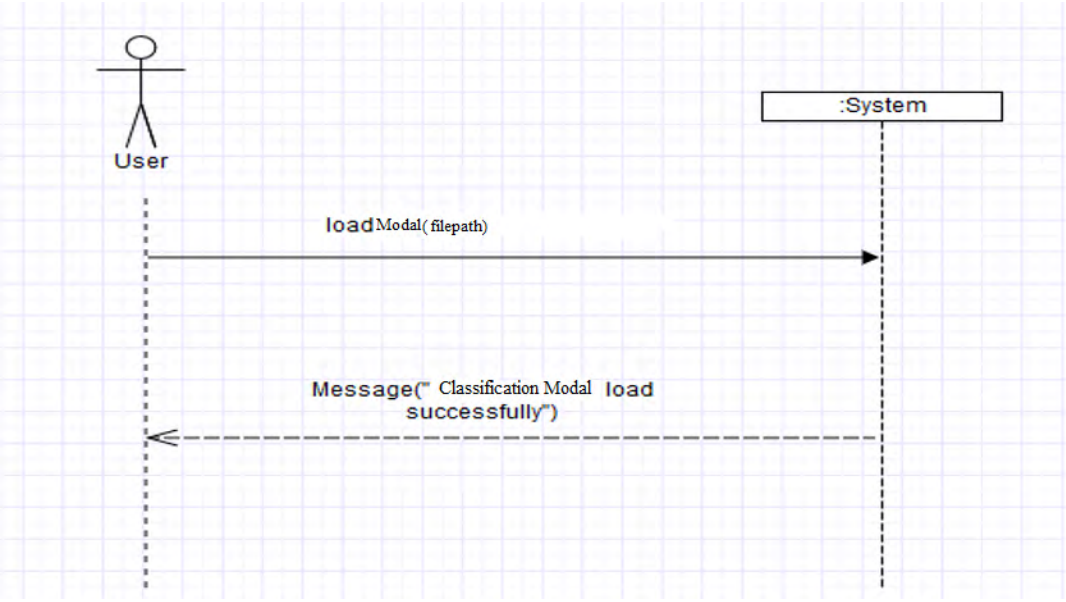


Figure 9 SSD for Load a Classification Modal

4.5.5 Annotate a video:

Following Figure shows the system sequence diagram of annotating a video:

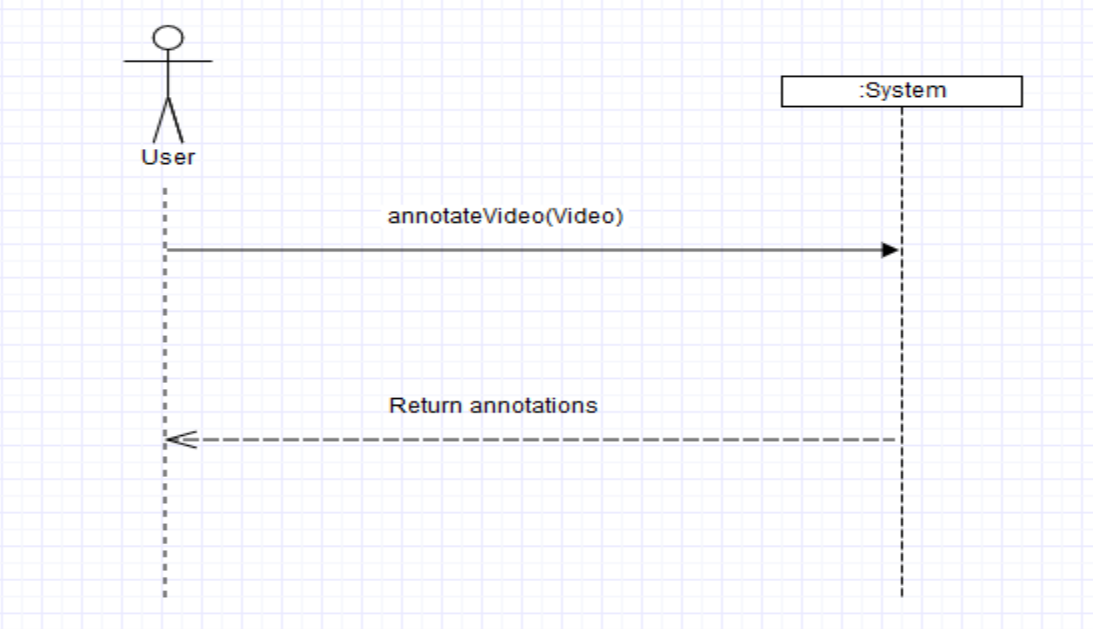


Figure 10 SSD for Annotate a Video

4.5.6 Save Annotations:

Following Figure shows the system sequence diagram of saving annotations:

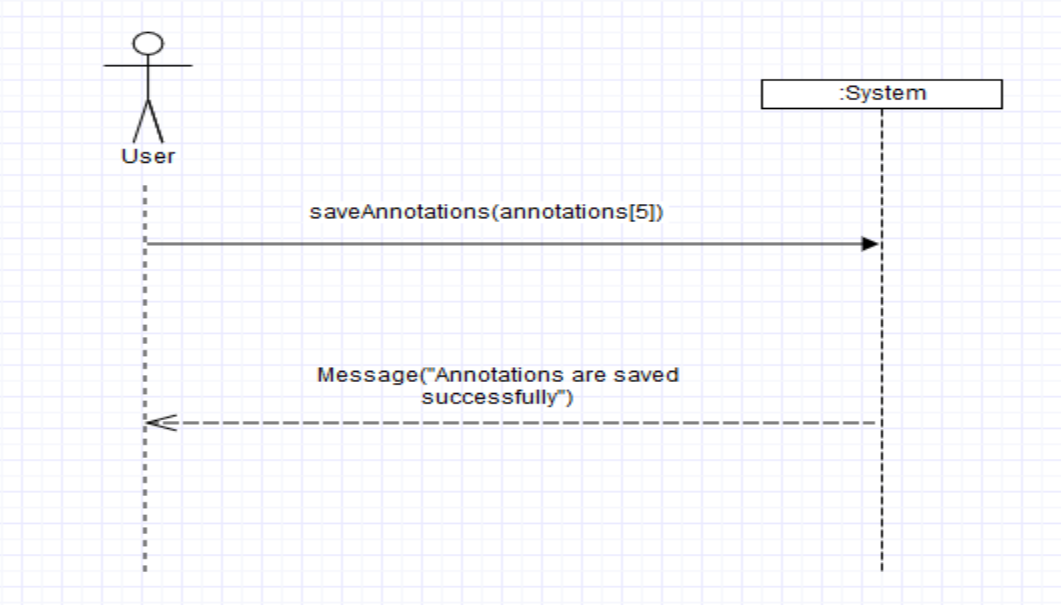


Figure 11 SSD for Save Annotations

4.5.7 Add Annotations:

Following Figure shows the system sequence diagram of adding annotations:

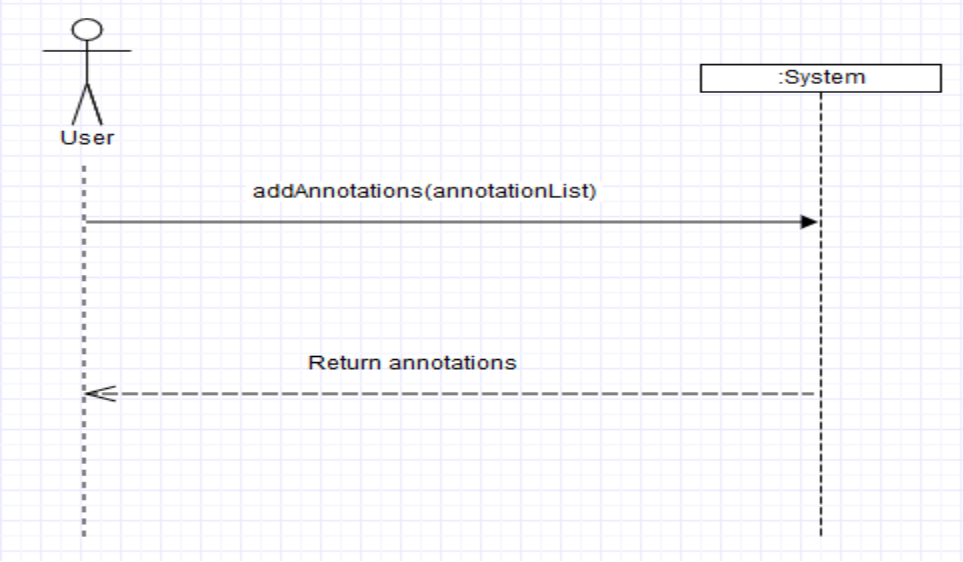


Figure 12 SSD for Add Annotations

4.5.8 Create dataset:

Following Figure shows the system sequence diagram of creating a dataset:

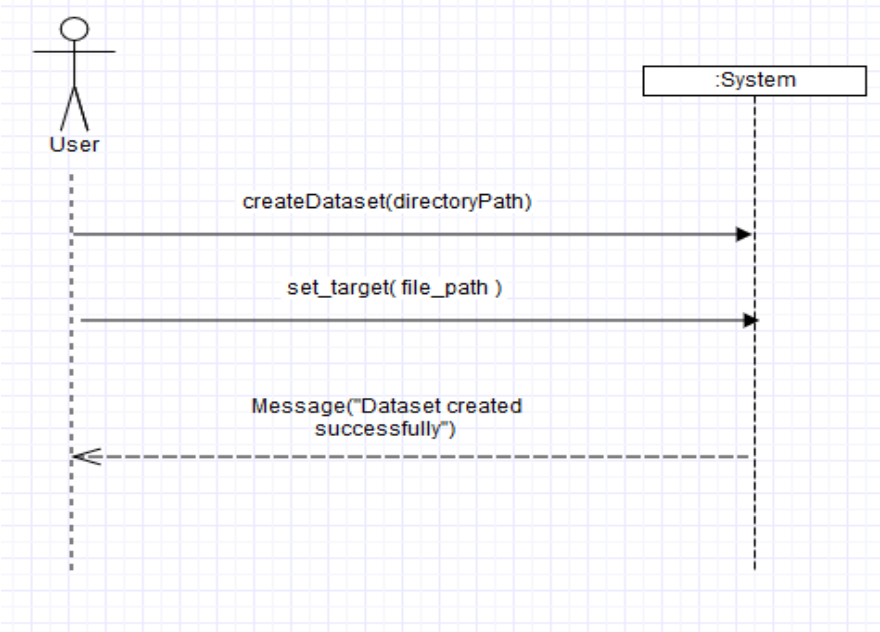


Figure 13 SSD for Create Dataset

4.6 Sequence Diagrams:

The order chart is used to present conversations between actors and objects in a system and in the conversation between objects themselves. A configuration chart shows the interactions that occur during a specific use case or use the case view.

4.6.1 Train a Classification Modal:

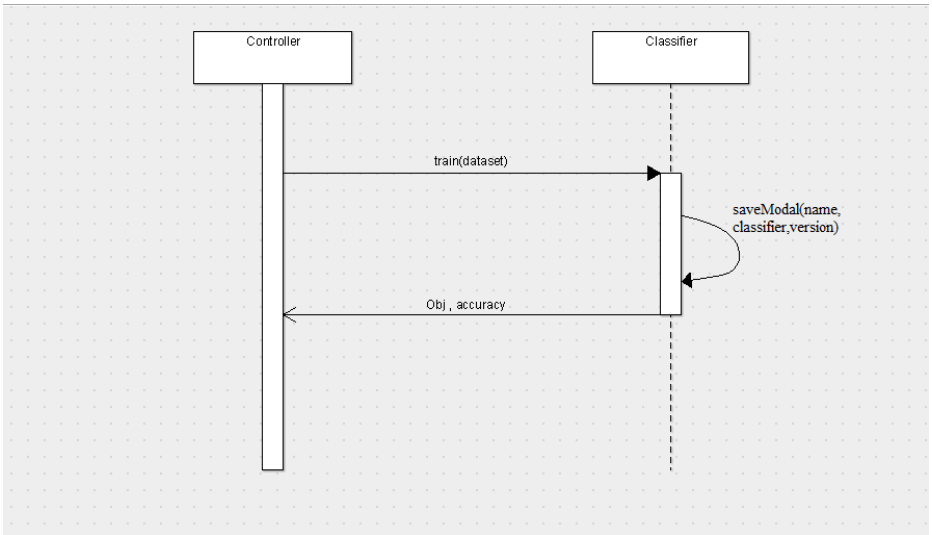


Figure 14 SD for Training a Classification Modal

4.6.2 Retrain a Classification Modal:

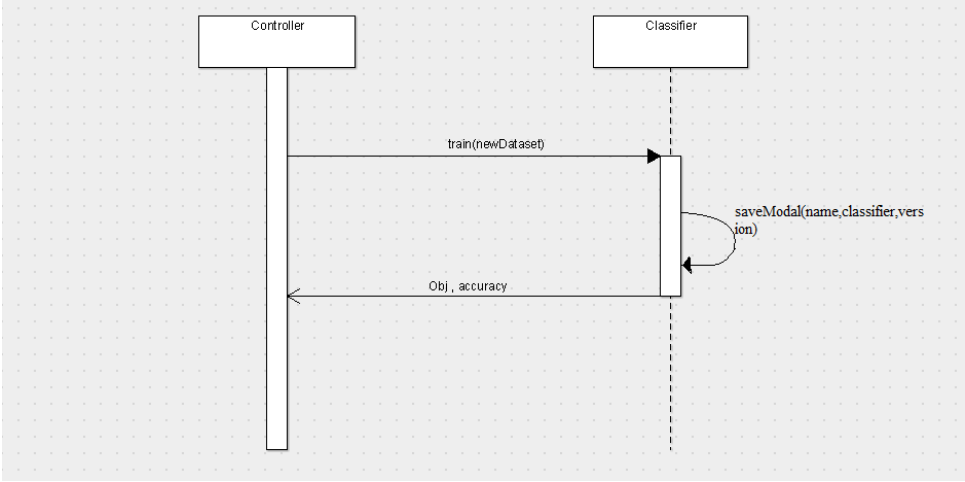


Figure 15 SD for Retrain a Classification Modal

4.6.3 Sava a Classification Modal:

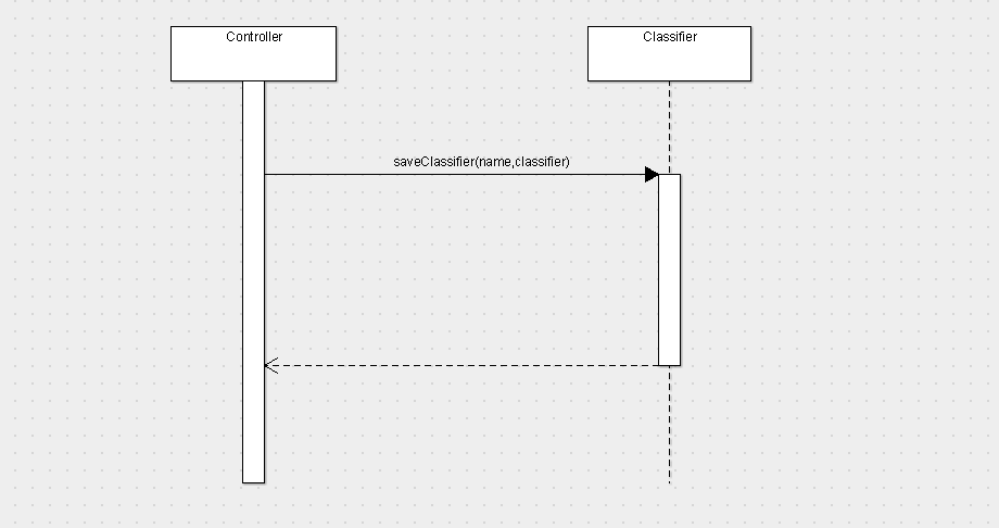


Figure 16 SD for Save a Classification Modal

4.6.4 Load a Classification Modal:

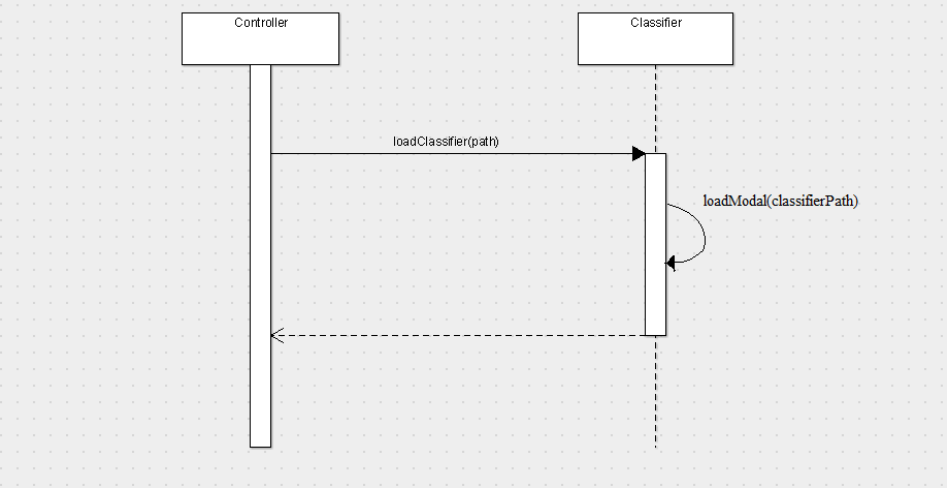


Figure 17 SD for Load a Classification Modal

4.6.5 Annotate a video:

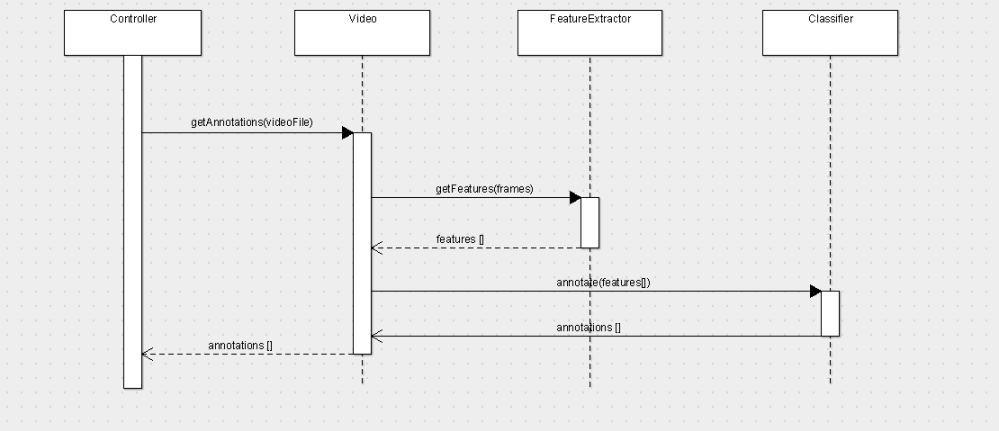


Figure 18 SD for Annotate a Video

4.6.6 Save annotations:

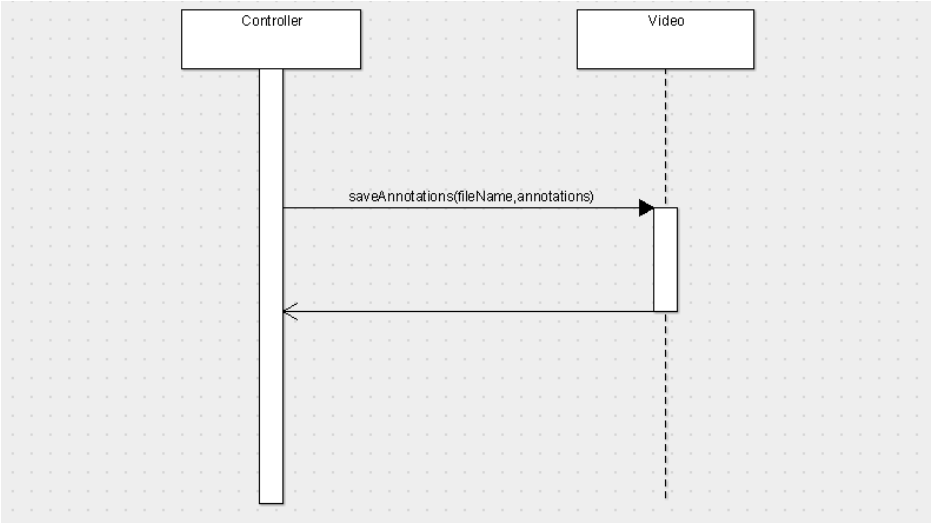


Figure 19 SD for Save Annotations

4.6.8 Create dataset:

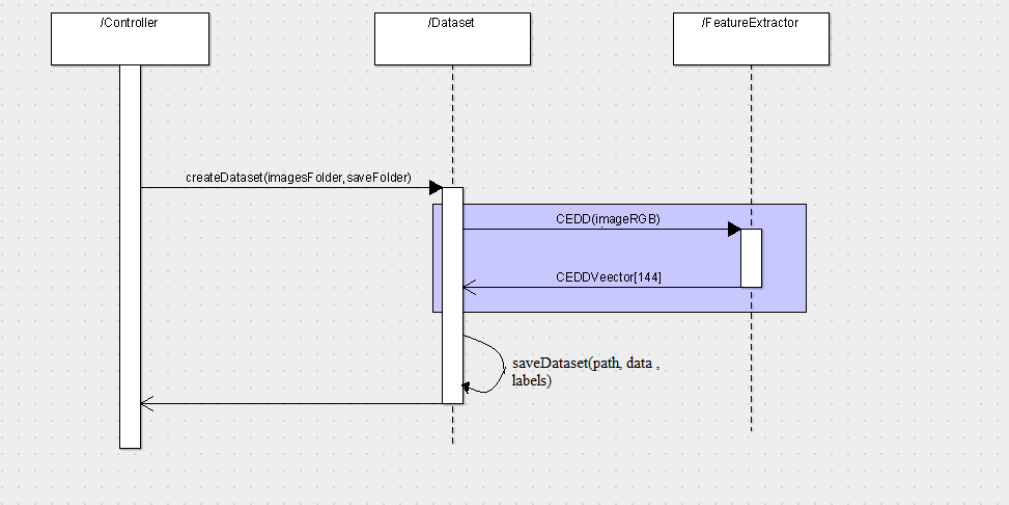


Figure 20 Create Dataset

4.7 Domain Model:

A domain model could be a visual illustration of the distortion of a field into individual subjective categories or objects. It is a way to describe and model real world entities and the relationships between them.

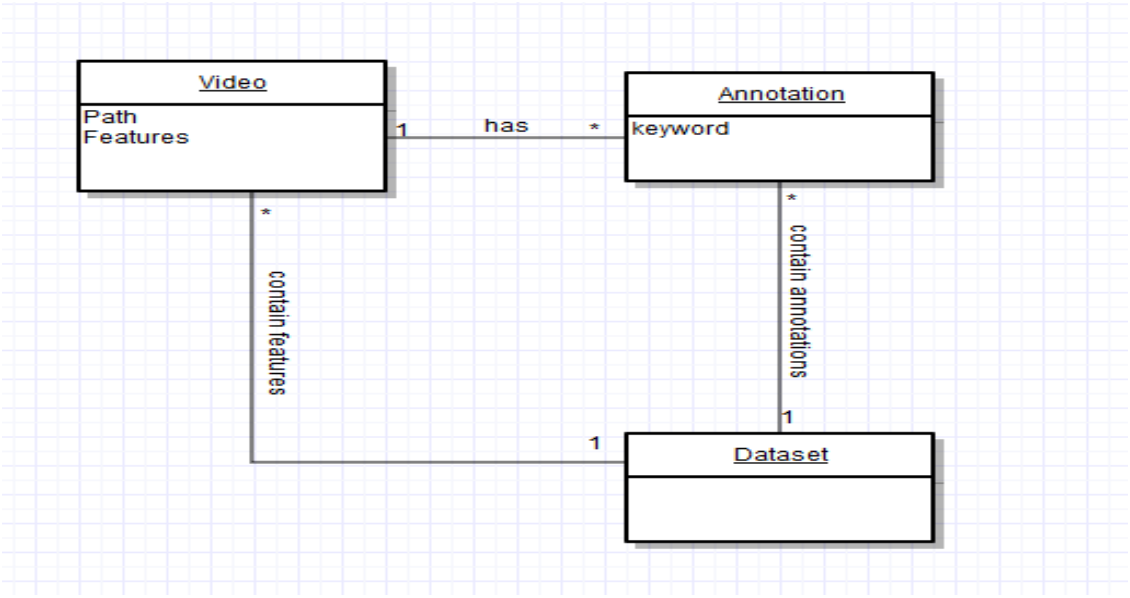


Figure 21 Domain Model

4.8 Class Diagram:

The Class diagram for Automatic Video Annotation Tool (AVAT) is as follows:

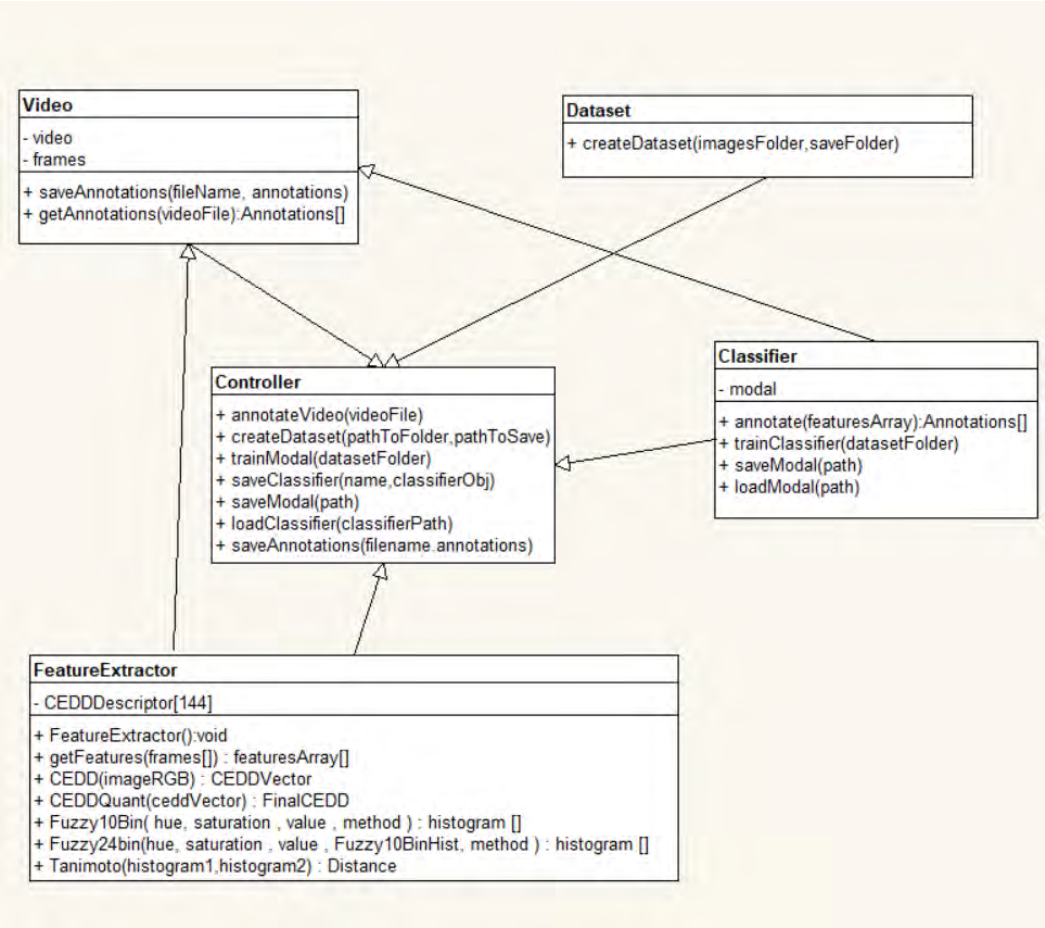


Figure 22 Class Diagram

4.9 User Interface Design:

User Interface (UI) style is the design of user interfaces for computer code or machines, in the same manner with the origin of the desktop application, focusing on ease of use and learning ability.

4.9.1 Interface for application:

Following are the interfaces for the application:

4.9.1.1 Save Classifier:

Interface for “Saving a Classifier” Use Case (UC) is shown below:

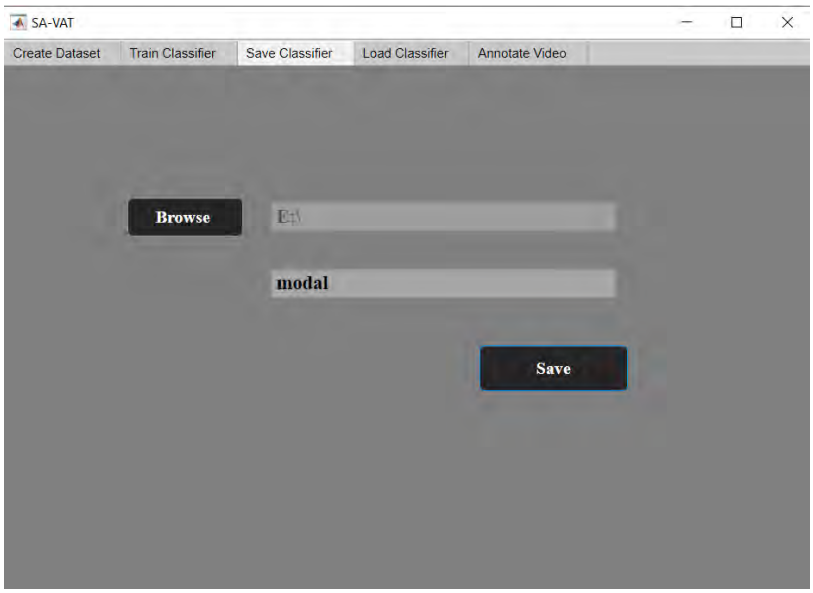


Figure 23 Interface for Save Classifier

4.9.1.2 Load Classifier:

Interface for “Loading a Classifier” Use Case (UC) is shown below:

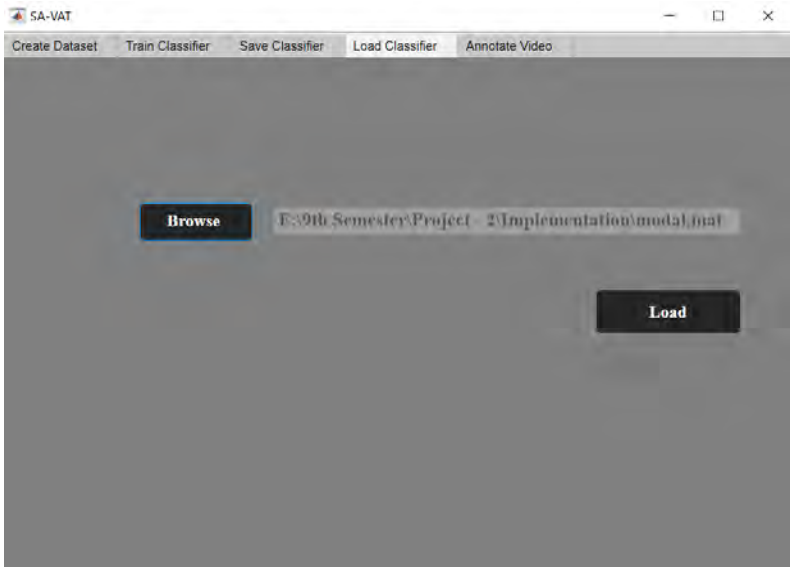


Figure 24 Interface for Load a Classifier

4.9.1.3 Train a Classification Modal:

Interface for “Train a Classification Modal” Use Case (UC) is shown below:

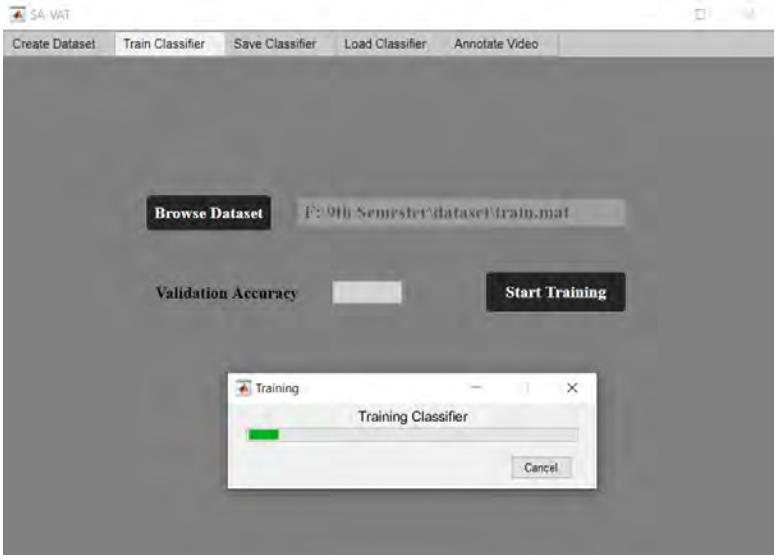


Figure 25 Train a Classification Modal

4.9.1.4 Retrain the Classification Modal:

Interface for “Retrain the Classification Modal” Use Case (UC) is same as training and is shown below:

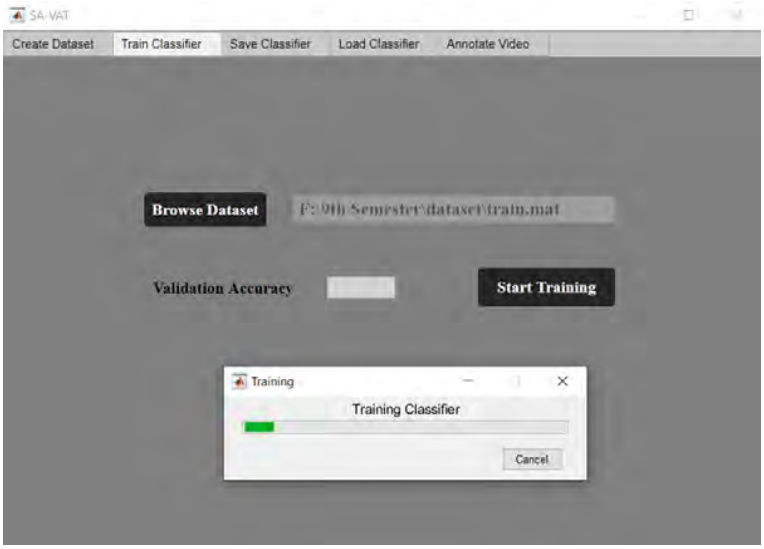


Figure 26 Interface for Retrain the Classification Modal

4.9.1.5 Create Dataset:

Interface for “Creating Dataset” Use Case (UC) is shown below:

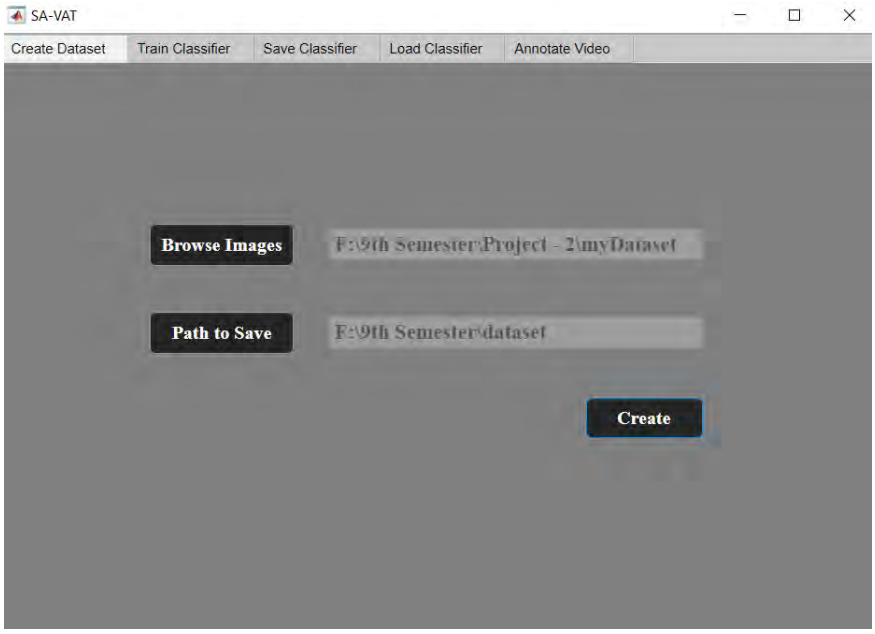


Figure 27 Create Dataset

4.9.1.6 Annotate Video:

Interface for “Annotate a video” Use Case (UC) is shown below:

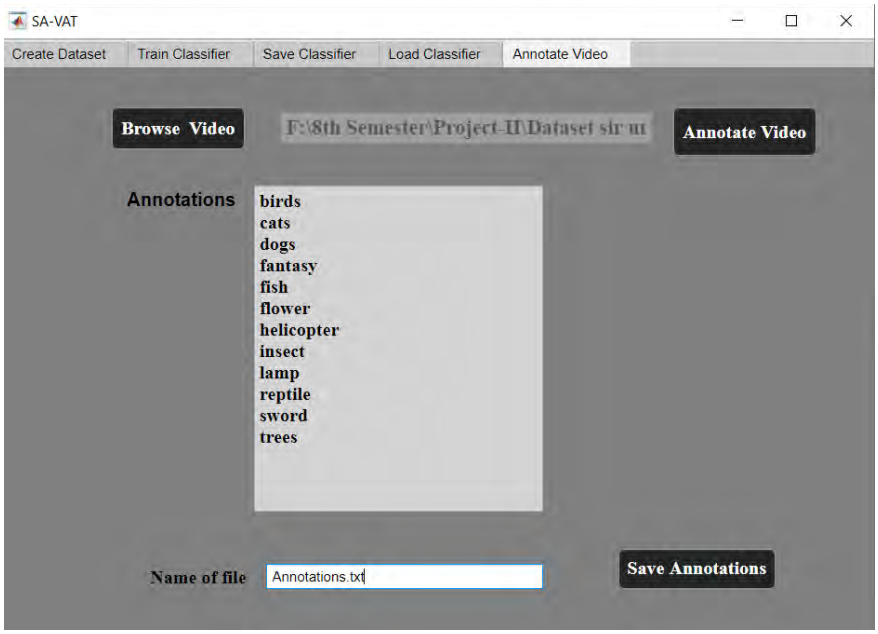


Figure 28 Interface for Annotate a Video

Chapter 5: Software Test Document

5 Software Test Document

5.1 Introduction:

A test case, in software engineering, is a set of conditions in which a tester will decide whether an application, software system, or one of its characteristics is actually being set up to do so.

5.1.1 Test Approach:

User Approval Testing (UAT) consists of a process to confirm a solution works for the user. You are not testing the system (the software does not crash and does not meet the requirements of the documents), but ensure that your solution will work for the user.

5.2 Test Plan:

A test plan is a document that explains the purpose, target market, and the process for specific tests for software or hardware products. This project usually involves a detailed understanding of the final work flow.

5.2.1 Features to be tested:

Features to be tested are as following:

1. Train a Classification Modal.
2. Retrain the Classification Modal.
3. Save a Classification Modal.
4. Load a Classification Modal.
5. Annotate a video.
6. Save annotation.
7. Add annotations.
8. Create dataset.

5.2.2 Features not to be tested:

Structures not to be tested are from the developer's point of view. For example

- How much memory is consumed by the tool?
- Software risk factor.
- Maintainability of the application.

5.2.3 Testing Tools & Environments:

A test environment is the installation of software and hardware for test teams to run test trials. In other words, it supports testing implementation with hardware, software and network configuration. Testing or testing environment is arranged as per the requirement for application under the test. This is where the black box test environment is used. The black box testing is a software testing method that evaluates the functionality of an application without the internal structure or pairing in operation. This test method can be applied to almost any level of software testing: drive, integration, system and acceptance.

5.3 Test cases:

Test cases for the applications are:

5.3.1 TC-01 Train a Classification Modal:

Following describes test case of "Train a Classification Modal":

Table 8 TC-01 Train a Classification Modal

Tester	User.
Purpose	User train a Classification Modal.
Setup	User has started the application/tool and select create dataset.
Inputs	User browse and select dataset successfully.
	User browse and didn't select dataset.
Steps	1. User selects the browse dataset option. 2. System display the list of directories. 3. User select the dataset directory. 4. Classification modal start training on the dataset and prompt user to wait.
Expected Result	Trained Classification Modal.
Actual Result	Trained Classification Modal.
Pass/Fail	Pass

5.3.2 TC-02 Retrain a Classification Modal:

Following describes test case of "Retrain the Classification Modal":

Table 9 TC-02 Retrain a Classification Modal

Tester	User.
Purpose	User retrain the Classification Modal.
Setup	User has started the application and classification modal is trained and saved before.
Inputs	User browse and select dataset successfully.
	User browse and didn't select dataset.
Steps	1. User selects the browse dataset option. 2. System display the list of directories. 3. User select the dataset directory. 4. Classification modal start the training on the new dataset and prompt user to wait.
Expected Results	Trained Classification Modal on a new dataset.
Actual	Trained Classification Modal on a new dataset.
Pass/Fail	Pass

5.3.3 TC-03 Save a Classification Modal:

Following describes the test case of “Save the Classification Modal”:

Table 10 TC-03 Save a Classification Modal

Tester	User
Purpose	User Save the Classification Modal for further classifying new examples.
Setup	User has started the application/tool and Classification Modal is trained.
Inputs	The user will select the "Save" option and give it a unique name.
	User select the “save” option and didn’t give name.
Steps	1. User selects the “save” option. 2. User selects the location where to save the classifier. 3. User gives unique name to the Classification Modal file. 4. Classification Modal is saved.
Expected results	Classification Modal is saved on the hard disk.
Actual	Classification Modal is saved on the hard disk.
Pass/Fail	Pass

5.3.4 TC-04 Load a Classification Modal:

Following describes the test case of "Load a Classification Modal":

Table 11 TC-04 Load a Classification Modal

Tester	User
Purpose	User loads an already trained and saved Classification Modal.
Setup	User has started the application/tool and has already trained Classification Modal saved on hard disk.
Inputs	User selects "Loads" option and browse for the file.
	User selects "Loads" option and didn't select the file.
Steps	1. User selects the "Load" options. 2. User browse the location where the Classification Modal file is saved. 3. User select the file. 4. System load the Classification Modal.
Expected results	Classification Modal is loaded into the system.
Actual	Classification Modal is loaded into the system.
Pass/Fail	Pass.

5.3.5 TC-05 Annotate a video:

Following describes the test case of “Annotate a video”:

Table 12 TC-05 Annotate a Video

Tester	User.
Purpose	User annotate a video.
Setup	User has started the application/tool.
Inputs	User selects the video to be annotated.
	User didn’t select any video.
Steps	1. User selects “Annotate” option. 2. User selects video to be annotated.
Expected results	System prompts the user with annotations.
Actual	System prompts the user with annotations.
Pass/Fail	Pass.

5.3.6 TC-06 Save annotations:

Following describes the test case of “Save annotations”:

Table 13 TC-06 Save Annotations

Tester	User
Purpose	User save the annotations.
Setup	User has started the application/tool and annotated a video.
Inputs	User gives a unique to file, in which annotations is to be saved and selects the directory where file is saved.
	User didn’t give a unique to file, in which annotations is to be saved or didn’t selects the directory where file is saved.
Steps	1. User give unique to file. 2. User selects the directory to save the file. 3. User select the “save annotation” option.
Expected results	Annotations are saved on hard disk.
Actual	Annotations are saved on hard disk.
Pass/Fail	Pass.

5.3.7 TC-07 Add annotations:

Following describes the test case of “Add Annotations”:

Table 14 TC-07 Add Annotations

Tester	User
Purpose	User add more annotations to the suggested ones.
Setup	User has stared the application/tool and has annotated a video.
Inputs	User enters more keywords.
	User didn’t enter any keyword.
Steps	1. User selects “Add annotations” option. 2. User enter more keywords. 3. User selects the “Add” option.
Expected results	More keywords are added to the original suggested ones.
	System prompts “Enter at least one keyword”.
Actual	
Pass/Fail	

5.3.8 TC-08 Create dataset:

Following describes the test case of "Create dataset":

Table 15 TC-08 Retrain a Classification Modal

Tester	User
Purpose	User create a new dataset.
Setup	User has started the application/tool.
Inputs	The user selects the video from computer and enter their annotations.
	The user selects the video from computer and didn't enter their annotations.
Steps	1. User selects the video. 2. User enter their annotations. 3. System extract their features and saved it with their annotations.
Expected results	New dataset is created.
Actual	New dataset is created.
Pass/Fail	Pass.

5.4 Evaluation:

Support Vector Machine classifier is being used for annotation suggestions of the system. Performance is measure by the accuracy of the classifier.

5.4 Experimental setup:

For evaluation of Classifier training state and performance, training and validation dataset is distributed using K-fold or Jackknife mechanism, which helps in overcoming overfitting problem. Dataset organization for 10 experiments is explained in the following table:

Table 16 Dataset Organization

Dataset	Total no. of instances	Training Instances	Validation Instances
DS1	25346	22812	2534
DS2	25346	22811	2535
DS3	25346	22811	2535
DS4	25346	22811	2535
DS5	25346	22811	2535
DS6	25346	22811	2535
DS7	25346	22812	2534
DS8	25346	22812	2534
DS9	25346	22812	2534
DS10	25346	22812	2534

Experiments are performed on all 10 datasets. Results of these experiments are listed in following table:

Table 17 Experiment Results

Experiment	Total Test instances	Training error	Average Training error
1	2534	0.221	0.2171
2	2535	0.229	
3	2535	0.218	
4	2535	0.223	
5	2535	0.215	
6	2535	0.207	
7	2535	0.211	
8	2534	0.226	
9	2534	0.209	
10	2534	0.212	

5.4.2 ROC curve:

A receiver operator feature **(ROC)** is a graphic plot used to demonstrate the ability to diagnose the Walker rating. Given. A ROC walker is built by the True Positive Rate (TPR) against False Positive Rates (FPR). The actual positive percentage is the percentage of observations that are correctly predicted to be positive from all positive observations (TP + FN). Similarly, all negative comments of false positive percentage observations (FP/ (TN + FP)) are positively predicted This mark shows false positive rates (FPR) and true positive rate (TPR) values.

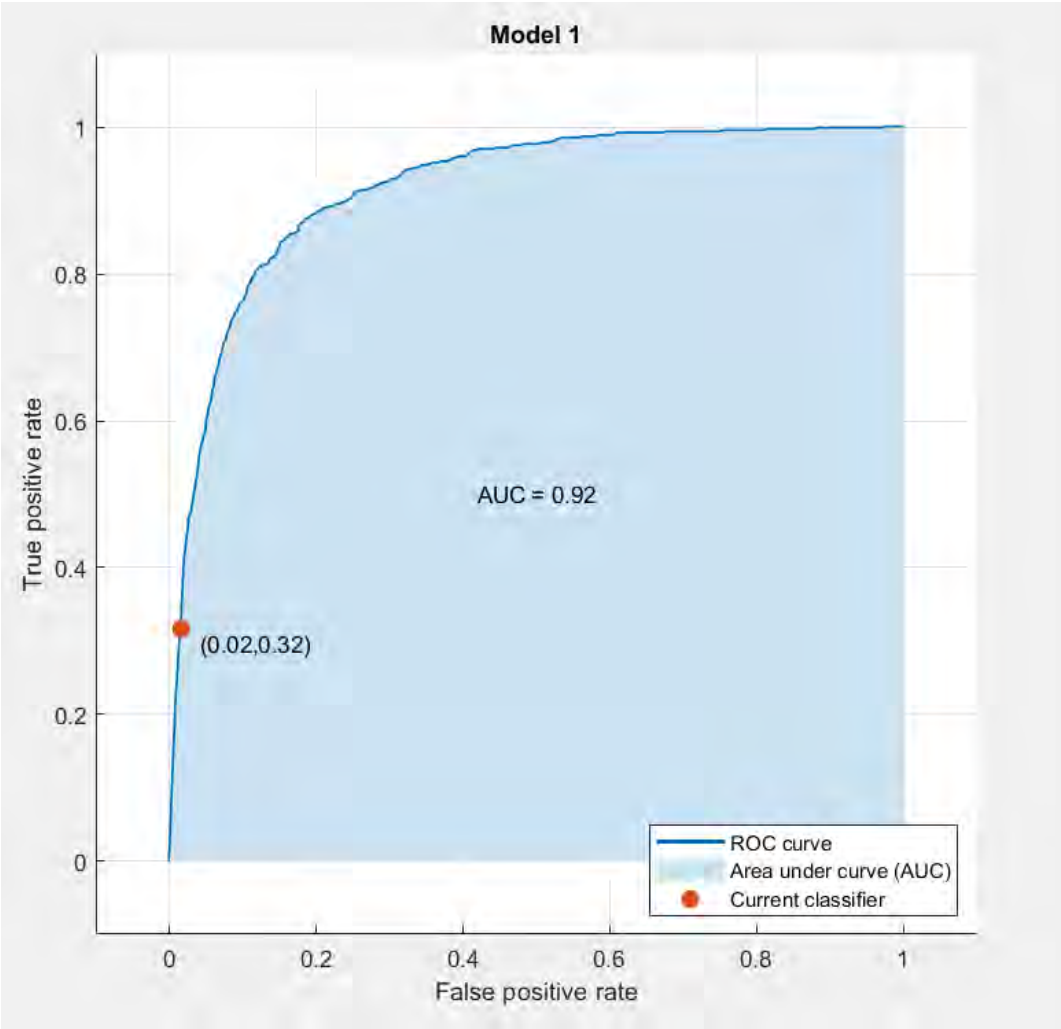


Figure 29 ROC Curve

5.4.3 Results and discussions:

An overview of the experiments, comparison of correctly classified and misclassified examples and actual error. Table 17 describes the 10-dataset organization of our dataset, which consists of 25346 instances of different objects.

Table 18 represents the experimental result of classifier. The training error is computed for every example.

5.5 Summary:

In section 5.1.1, testing technique is described. In the second part, evaluation and performance of the classifier is discussed using kfold.

References:

[1] Carl Vondrick, Donald Patterson, Deva Ramanan. “Efficiently Scaling up Crowdsourced Video Annotation” International Journal of Computer Vision (IJVC). June 2012.

[2] T.A. Biresaw, T. Nawaz, J. Ferryman, A. Dell, “ViTBAT: Video Tracking and Behavior Annotation Tool”, IEEE International Conference on Advanced Video and Signal Based Surveillance, Colorado, Springs, USA -2016.

[3] Scalabel Tool https://github.com/scalabel/scalabel.

[4] Thesis Shen, Anting BeaverDam: Video Annotation Tool for Computer Vision Training Labels EECS Department, University of California, Berkeley 2016 December 8 UCB/EECS-2016-193 Shen: EECS-2016-193

[5] LabelMe: a database and web-based tool for image annotation. B. Russell, A. Torralba, K. Murphy, W. T. Freeman. International Journal of Computer Vision, 2007.

[6] Ambardekar, Amol et al. “Ground Truth Verification Tool (GTVT) for Video Surveillance Systems.” 2009 Second International Conferences on Advances in Computer-Human Interactions (2009): 354-359.

[7] J. Jäger: Open hyper video in REMEDIATE – at the Borders of Film, Internet and Archives, M. Doulis and P. Ott, Ed. München: Wilhelm Fink Verlag, 2013, pp. 228-269.

[8] Wazalwar, Sampada S. and Latesh G. Malik. “A Survey on Video Annotation Techniques.” (2013).

[9] Motaz El-Saban, Xin-Jing Wang, Noran Hasan., “Seamless annotation and enrichment of mobile captured video streams in real time”, IEEE 2011

[10] Xingquan Zhu, Jianping Fan, Xiangyang Xue, Lide Wu and Ahmed K. Elmagarmid, “ Semi-automatic video content annotation”, Proceeding of Third IEEE Pacific Rim Conference on Multimedia, pp. 37-52, 2008.

[11] Takuho Nakano, Akisato Kimura, “Automatic video annotation via hierarchical topic trajectory model considering cross-modal correlations” IEEE 2011.

[12] Xu Zhao, Kai-Hsiang Lin, Yun Fu, “Text From Corners: A novell approach to detect text and caption in videos”, IEEE transaction on image processing, vol..20, No. 3, MARCH 2011.

[13] Shih-Wei Sun, 2011.Yu-Chiang Frank Wang, “Automatic annotation of web videos”, IEEE 2011.

[14] Shih-Wei Sun, 2011.Yu-Chiang Frank Wang, “Automatic annotation of web videos”, IEEE 2011.

[15] M. Jacob, Blu and M. Unser, “Efficient energies and algorithms forparametric snakes”, IEEE Transactions on Image Processing; vol. 13, no.9, pp.1231-1244, 2004.

[16] Messing, D. S. ; van Beek, P.; Errico, J. H., “The MPEG-7 color structure descriptor: image description using colour and local spatial information”, Proceedings of International Conference on Image Processing, pp.670-673, October 2001.

[17] Chee Sun Won, Dong Kwon Park, and Soo-Jun Park., “Efficient use of Mpeg- edge histogram descriptor”, ETRI Journal 2002; vol. 24, no. pp.23-30.

[18] Tianzhu Zhang, Changsheng Xu, Guangyu Zhu, “A Generic Framework for Video Annotation via Semi-Supervised Learning”, IEEE TRANSACTIONS ON MULTIMEDIA, VOL. 14, NO. 4, AUGUST 2012.

[19] Yu-Gang Jiang, Qi Dai, Jun Wang, Chong-Wah Ngo, “Fast Semantic Diffusion for Large-Scale Context-Based Image & Video Annotation”, 3080 IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 21, NO. 6, JUNE 2012.