MOOCs RECOMMENDER BASED ON LEARNING STYLES

Software Requirement Specification

Project ID: 18-036

Author: Saugat Aryal

IT 14146602

Bachelor of Science Special (Honors) in Information Technology

Specializing in Software Engineering

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

Date of Submission: 2018-05-15

MOOCs RECOMMENDER BASED ON LEARNING STYLES

Project ID: 18-036

Author: Saugat Aryal

IT 14146602

Supervisor: Mr. Nuwan Kodagoda

Date of Submission: 2018-05-15

DECLARATION

I hereby declare that the project work entitled “MOOCs Recommender Based on Learning Styles” (Video Style Classification component) submitted to the Sri Lanka Institute of Information Technology, is a record of original work done by our group under the guidance of Mr. Nuwan Kodagoda (Supervisor) and Ms. Kushnara Suriyawansa (Co- Supervisor), and this project work is submitted in the fulfillment for the award of the Bachelor of Science (Special Honors) in Information technology Specialization in Software Engineering. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma. The diagrams, research results and all other documented components were developed by myself and I have cited clearly any references I have made.

Name: Saugat Aryal Signature:

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Purpose

The purpose of this document is to present a detailed description of requirements for Video Style Classification Component of our research project, MOOCs Recommender Based on Learning Styles. The document will explain the purpose and features, the interfaces, what the system will do and the constraints under which it must operate. All parts are intended primarily for customers of the application but will also be of interest to software engineers building or maintaining the software.

1.2 Scope

This document is intended for both stakeholders and the developers of the system. This document covers the requirements for the initial release of Video Style Classification Component of the proposed “MOOCs Recommender Based on Learning Styles” application. The application will work as a platform to provide the most suitable, relevant and optimal learning resources to the learners based on their preferred style of learning. The above-mentioned component deals with two sub-tasks: identifying various video lecture styles contained in a MOOC video and their level of composition and finally calculating the aggregate for the overall course. The main objective of classifying and determining the aggregate of different video styles is such that it can provide essential information when mapping MOOCs with the learning style. This document will focus on addressing the requirements to develop the component.

1.3 Definitions, Acronyms and Abbreviation

MOOCs	Massive Open Online Courses
CNN	Convolutional Neural Network
OS	Operating System
RAM	Random Access Memory
SRS	Software Requirement Specification
GPU	Graphics Processing Unit

1.4 Overview

The final product is targeted to be used by any learner who wants to learn using MOOCs. The learners can use the system to find the most appropriate and suitable MOOC courses based on their learning styles. Video Style Classification is a sub-component of the overall system and its main objective is to calculate the aggregate composition level of different styles used throughout the course.

This SRS document intends to cover all the functional and non-functional requirements of the Video Style Classification Component. Each of them has been discussed clearly in detail. All are described under three chapters. The first chapter provides an overall description of the component. The purpose and scope of the SRS is also mentioned.

The second chapter will provide an in-depth overview of the functionalities as focusing on users. The section will focus on comparing the software application to the existing systems in the market. It then moves on to explain several interfaces, constraints and operation of users to provide the reader with a better perspective of understanding the product. The chapter then explains about the summary of major functions, characteristics of users, constraints of system, assumptions and dependencies and finally conclude with apportioning of requirements.

The third chapter includes the specific requirements of the system and will be written primarily for developers. This section will describe the technical aspects and in-depth detail of the functionalities mentioned in the previous chapter. Both sections of the document describe the same software product in its entirety but are intended for different audiences and thus use different languages. The document will conclude with providing any supporting information regarding the content of the document.

2 OVERALL DESCRIPTIONS

Video lectures are the fundamental and significant component of MOOCs. They are produced in various standard styles that are used across different MOOCs platforms (edX, Coursera and Futurelearn). Usually, a single video is the composition of these different styles. Guo et al.[1] mentions 6 different types of production styles: Slides (PowerPoint presentation with voice-over), Code (video screencast of writing code), Khan-style (full-screen video of instructor drawing free-hand), Classroom (video captured in live classroom), Studio (recording in a studio with no audience) and Office Desk (close-up shots of instructor at an office desk). Similarly, Hansch et al.[2] also presents similar video styles using different names. Other additional styles like, Animation, Conversation, Text-Overlay, Picture-in-Picture are also described.

Video Style Classification component rely on this knowledge to identify different styles used in a video and calculate the quantity of each style in measurable unit (%). Finally, a net value for each style can be determined for the overall course by calculating the mean or average.

2.1 Product Perspective

The popularity of videos is increasing exponentially because of its usage in multiple platforms. Also, with the advancement in deep learning, CNN(Convolutional Neural Networks) and computational power, research in video classification techniques and algorithms has spiked up. Initially, the standard approach to video classification included features extraction and passing it to an SVM classifier to distinguish the classes. [3] proposed CNNs as powerful class of models for video classification. The video is treated as a bag of short, fixed-sized clips. The authors also describe a multiresolution for addressing the computational efficiency. Comparison of various

video classification techniques is presented in [4] and a new approach recognized as ‘Long Short Term memory Recurrent Neural Network’ is proposed to increase the performance of video classification with better time and space complexity. Two different methods: Feature Pooling and LSTM, capable of aggregating frame-level CNN outputs in video-level predictions is proposed in [5].

A similar work of classifying an educational video frame into a particular category is presented in [6]. The proposed system is composed of two components: Feature extractor and Feature processor. Feature extraction is carried out using AlexNet and the feature processing/classification is performed through SVM classifier. A given video was not classified into any category rather than being incorrectly classified was the most common failure. Another limitation identified was that the system classifies based on individual video frames whereas taking few adjacent video frames would be easier and efficient.

2.1.1 System Interfaces

- Database connectivity interface
- Web Scraping interface

2.1.2 User Interfaces

The video classification component is a backend process. Hence, no interfaces to interact with the user.

2.1.3 Hardware Interfaces

No special hardware devices are needed other than the usual PC or Laptop.

2.1.4 Software Interfaces

- Google Colaboratory
- Jupyter Notebook
- Google Tensorflow
- Keras – Deep Learning Library

2.1.5 Communication Interfaces

The user's laptop/desktop should support high speed data communication methods such as 3G (HSPDA) or 4G (LTE).

2.1.6 Memory Constraints

- RAM of 6 GB or higher

2.1.7 Operations

Functionalities of the Video Classification Component are carried out in the backend and hence, no operations are required by the user.

2.1.8 Site Adaption Requirements

Since the user can be of any nationality the user interface must be created for English language.

2.2 Product Functions

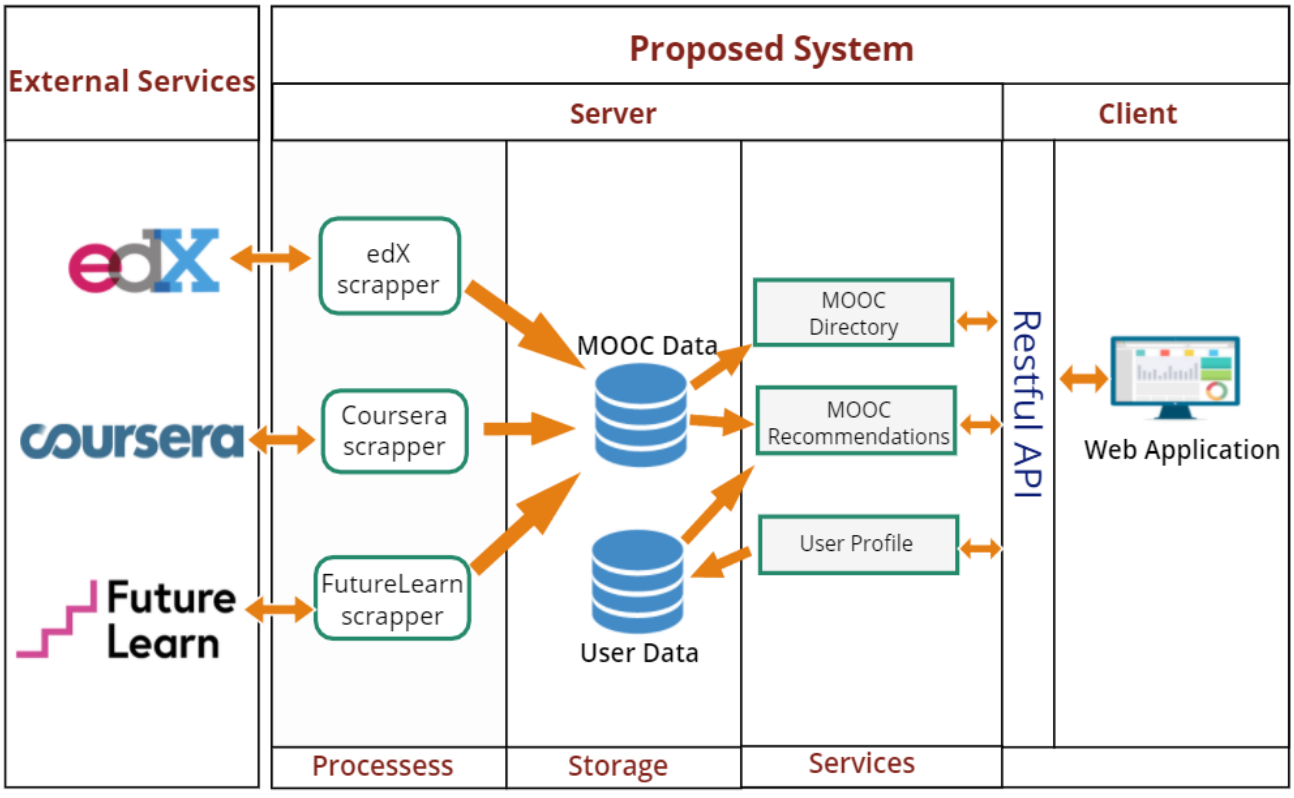

Figure 1: High Level System Architecture

In this component, the videos are first decomposed into a set of image frames. Before the image is sent through the trained CNN, it goes through a pre-processing step where the dimension of the image is reduced and transformed into low resolution. Finally, it passes through the network and is classified into one of the video style categories. The process is repeated for all the fames and all the videos. Finally, the amount of composition for each video style is calculated for a single MOOC video and also for the overall course.

2.2.1 Use Case Diagram

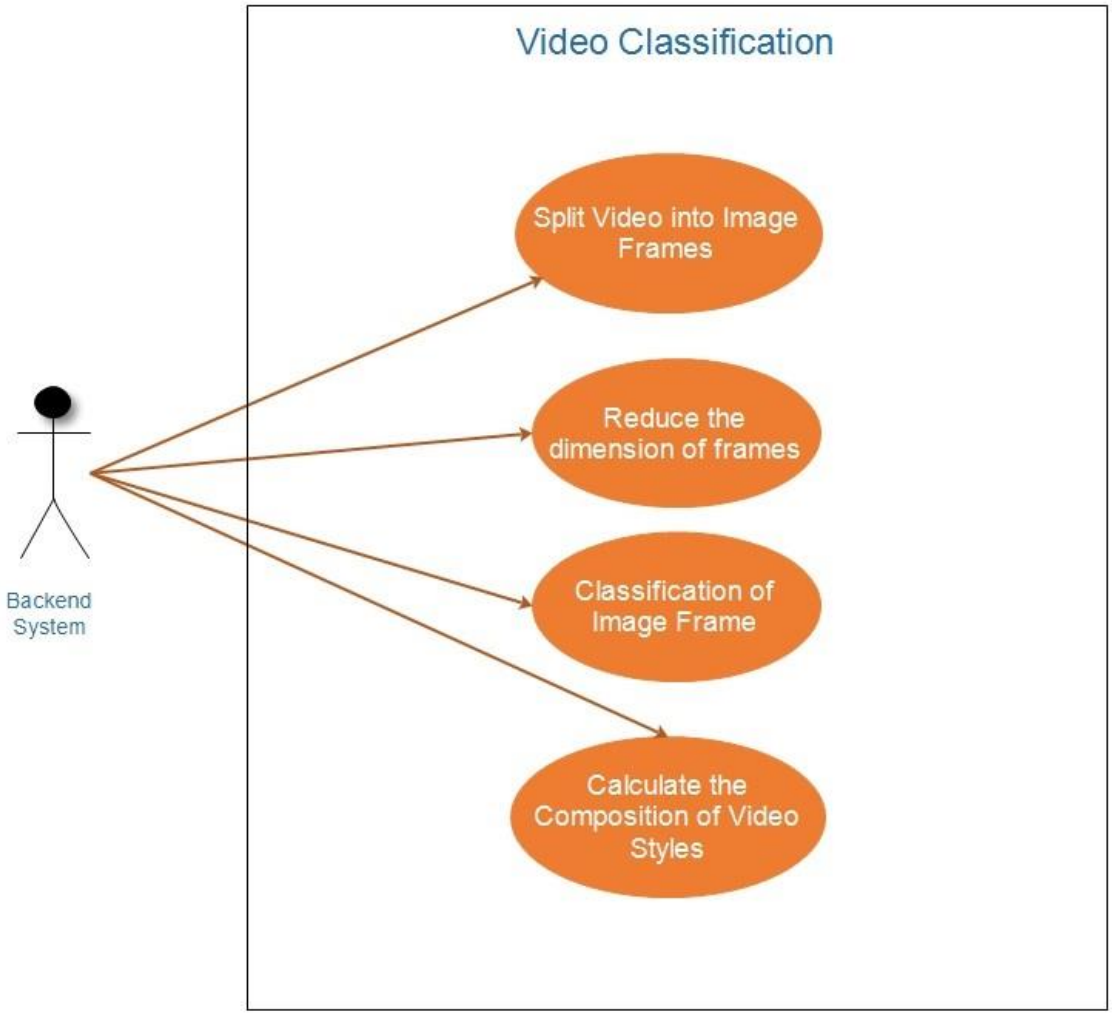

Figure 2: Use Case Diagram

2.2.2 Use Case Scenarios

Use case Name	Split Video into Image Frames
Pre –Condition	Videos are downloaded from the platforms
Post-Condition	Collection of Image Frames are available
Actor	Backend System
Main Success Scenarios	1. Select the video from the stored location. 2. Decompose into set of frames. 3. The use case ends with successfully splitting the video file into set of image frames.
Extension	1a. Invalid video source file is selected. 1b. Select a valid source file and proceed.

Table 1: Use Case Scenario 1

Use case Name	Pre-processing of Image Frames
Pre –Condition	Video is decomposed into Image Frames
Post-Condition	Image Frames with reduced size and dimension
Actor	Backend System
Main Success Scenarios	1. Select the image frames from the stored location. 2. Transform the image into lower dimension and graphics. 3. The use case ends with successfully converting the image frame into reduced dimension and properties.
Extension	1a. Invalid image frame is selected. 1b. Select a valid source file and proceed.

Table 2: Use Case Scenario 2

Use case Name	Classification of Image Frame into Video Style
Pre –Condition	Image Frames are transformed into lower dimension
Post-Condition	Categorization of Image Frame
Actor	Backend System
Main Success Scenarios	1. The image frames are passed through trained CNN. 2. The neural network classifies the image frame into one of the video style class. 3. The use case ends with, the model predicting the video style of the image frame.

Table 3: Use Case Scenario 3

Use case Name	Calculate the Composition of Video Styles
Pre –Condition	Image Frames are successfully classified by the model
Post-Condition	The amount of each video styles in the course
Actor	Backend System
Main Success Scenarios	1. Group the similarly classified video styles together. 2. Calculate the composition level of video styles for a single video in percentage. 3. Using Mean function or other aggregate function, calculate the composition of video styles for the overall course. 4. The use case ends with successfully storing the calculated details in the database.

Table 4: Use Case Scenario 4

2.2.3 Activity Diagram

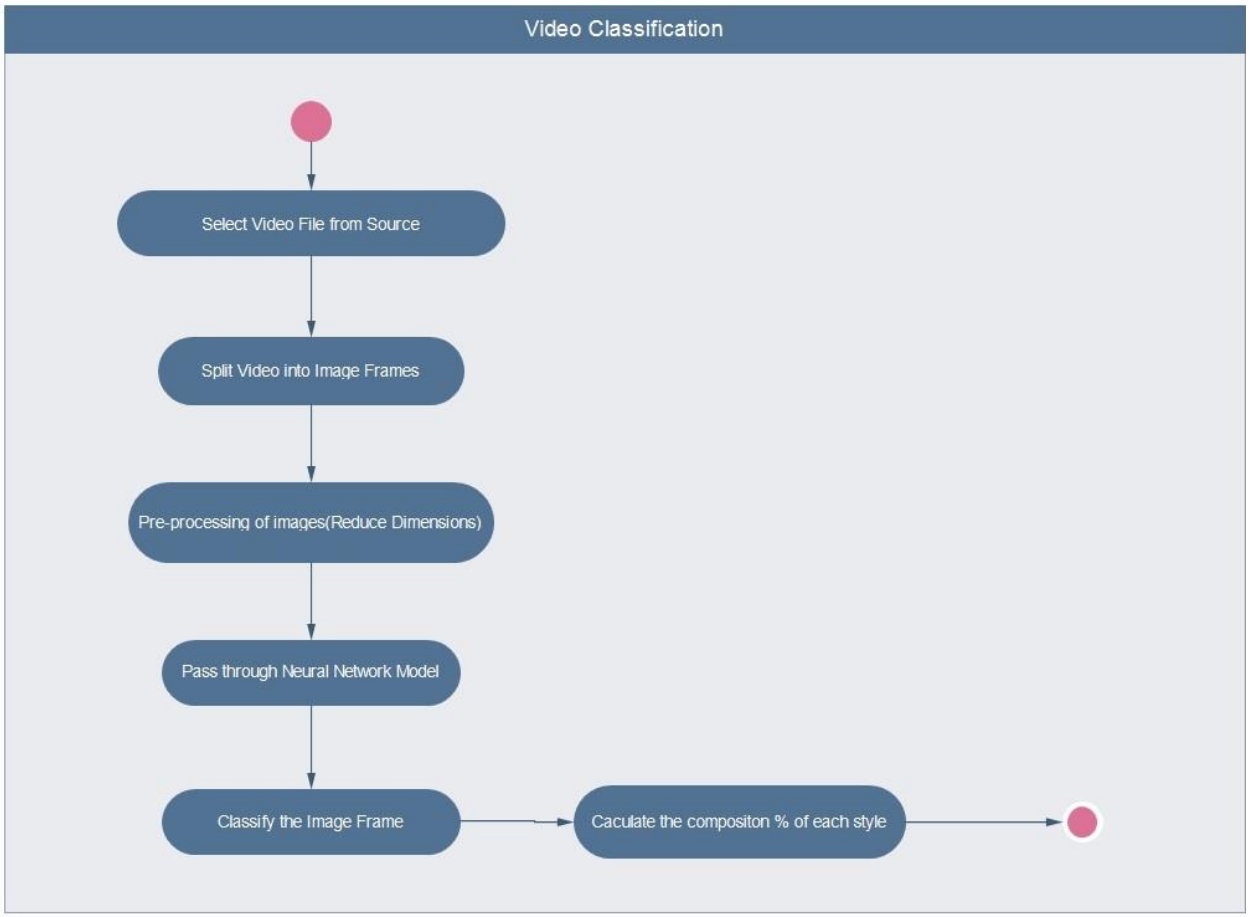

Figure 3: Activity Diagram

2.3 User Characteristics

The video classification component deals with the backend process and involves developing a neural network to classify the video style of the decomposed video image frames. It also consists of initial pre-processing steps and calculations at the later part, the user should be a software professional with appropriate knowledge of CNN and other deep learning tools and technologies including basic mathematics.

2.4 Constraints

The backend process of video classification involves heavy computational tasks. The constraints of the process are mentioned below:

- Backend system must have minimum of 6GB RAM. Additional 2GB of GPU will be advantageous.
- Windows 8 or higher.
- Python is the implementation language.
- Keras – Deep learning library is used to build the model.
- TensorFlow is used as the computational framework.

2.5 Assumptions and Dependencies

- The operating system for the video classification component is selected to be Windows.
- The internet connectivity can be established with ease on request.

2.6 Apportioning of Requirements

Essential Requirements:

1. Take a MOOC video and split it into set of image frames.
2. Perform pre-processing to reduce the dimension and size of the frames.
3. Pass the frames into the trained CNN model to classify into one of the video style categories.

4. Calculate the composition percentage of each video style for a single video and as well as for the overall course.

Desirable Requirement:

1. Visualize the results graphically to generate analysis and reports.

3 SPECIFIC REQUIREMENTS

3.1 External Interface Requirements

3.1.1 User Interfaces

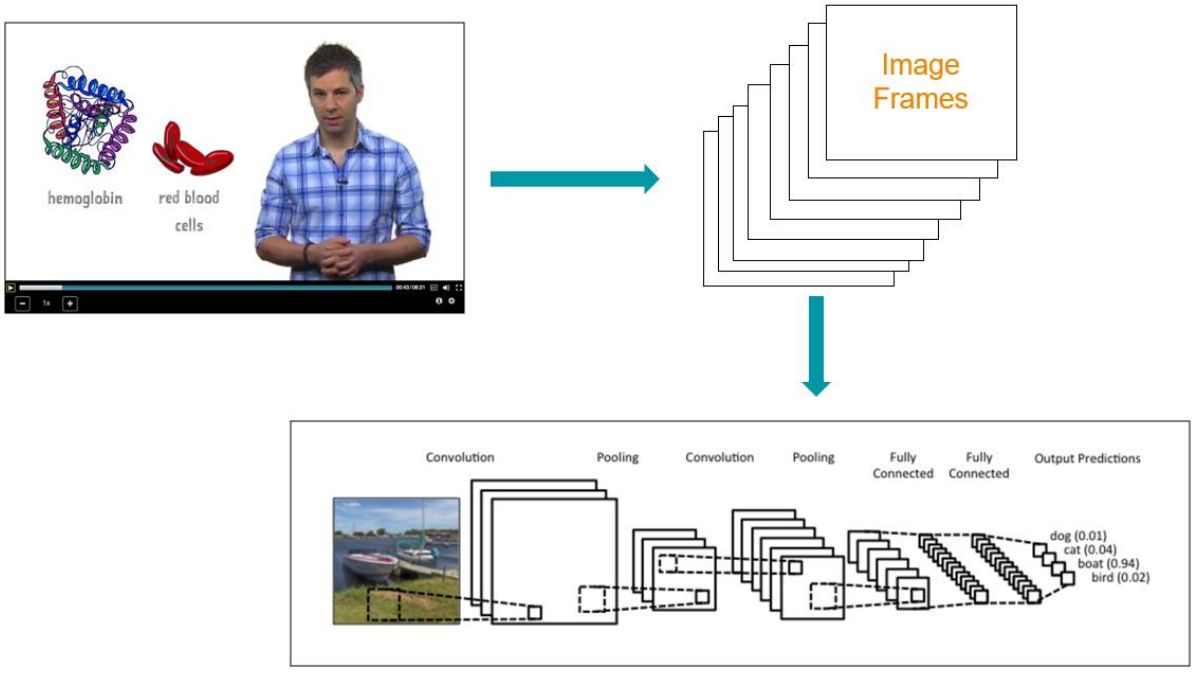

Figure 4: Video Splitting and Passing through CNN

The above figure gives a brief idea regarding the overall process. Given a MOOC video, it is first split into set of frames and passed through CNN which classifies it into one of the video style category

The training dataset has to be manually generated for different video styles. Sample example of datasets are:

Figure 6: Talking Head

Figure 5: Picture-in-Picture

Figure 8: Presentation Slide with Voice Over

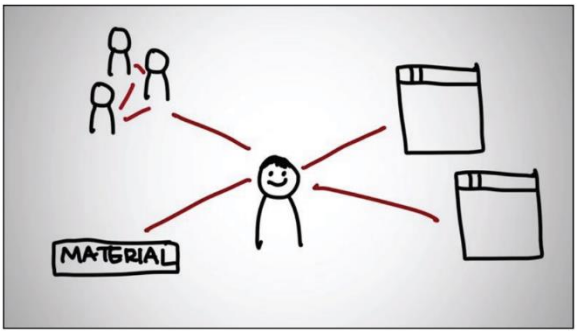

Figure 7: Animation

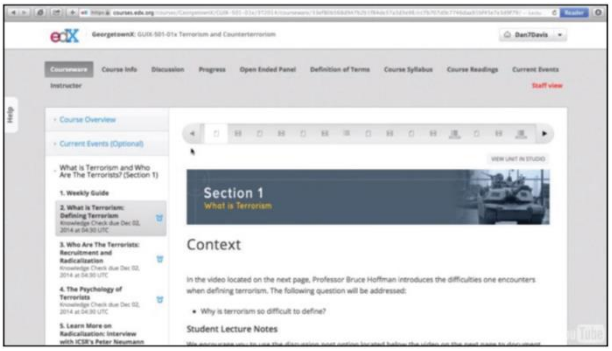

Figure 10: Tutorial/Demonstration

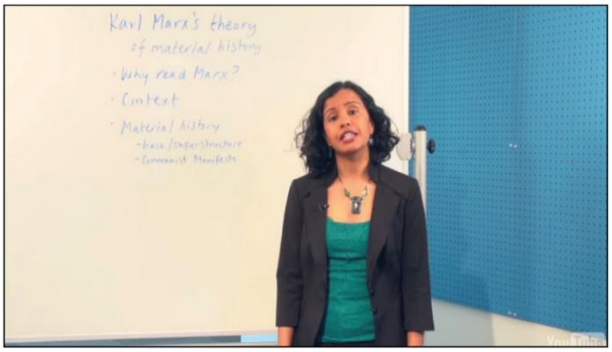

Figure 9: Whiteboard

After the video frames are passed through the CNN model, it classifies into one of the above video styles.

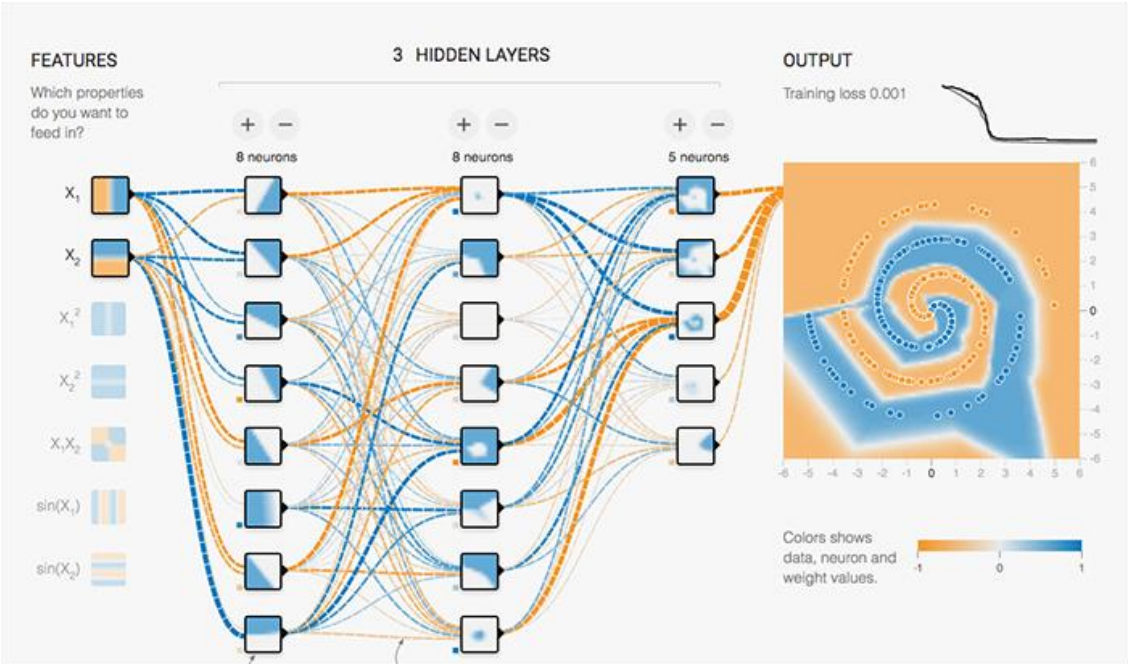

Figure 11: Neural Networks in TensorFlow

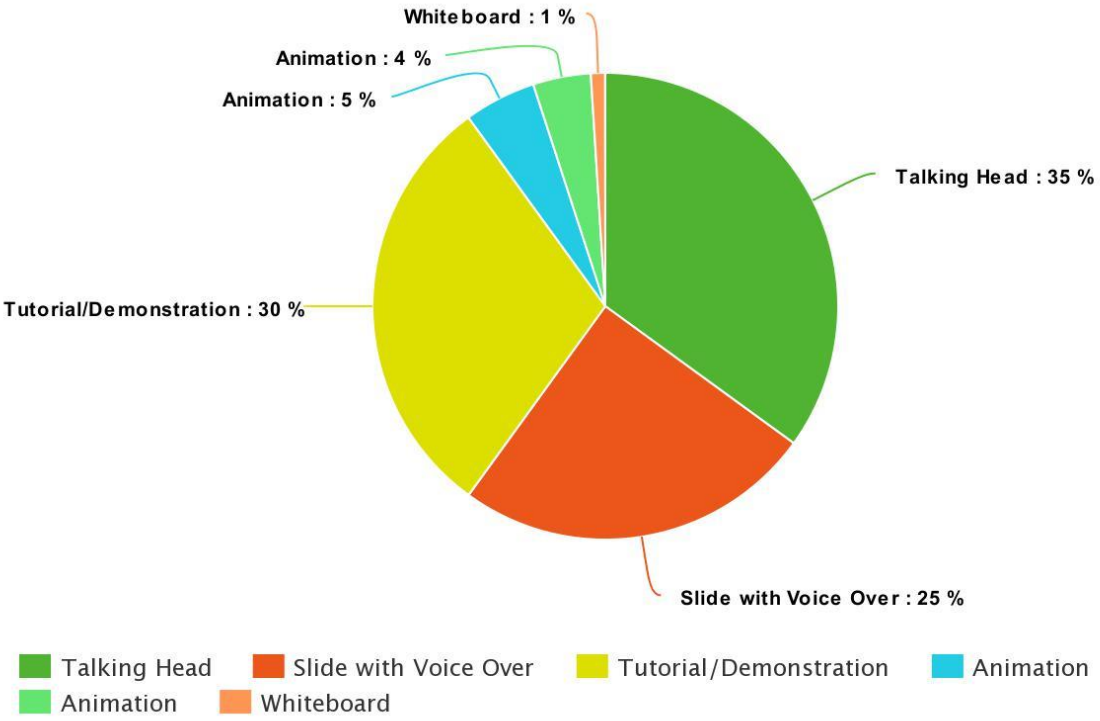

Figure 12: Statistics of Video Styles

After the video styles are classified by the model, aggregate values can be determined for the overall course and statistics can be generated as shown in figure 10.

3.1.2 Hardware Interfaces

- Graphics Processing Unit (GPU) – It can be used to speed up the computational tasks of training a neural network. It is all about leveraging the parallelism to perform mathematically compute intensive operations that CPU is not designed to handle. Training a CNN model can take days and even weeks when carried out in normal CPU, hence GPU can help to reduce the heavy load.

3.1.3 Software Interfaces

- Google Colaboratory: It is free cloud service offered by Google that provides free GPU. That distinctive feature makes it different than other cloud services. Using this service, users can develop deep learning applications using popular libraries such as, Keras, TensorFlow, OpenCV etc.
- TensorFlow: It is a computational framework for building machine learning models. It provides wide variety of toolkits that allows the users to construct models at the preferred level of abstraction. It comes in-built with Google Colab.
- Keras: It is a very popular deep learning library written in Python. We can quickly build and test a neural network with minimal lines of code. It is capable of running on top of TensorFlow and also available in Google Colab.

3.1.4 Communication Interfaces

The backend service along with the database is hosted on the cloud. Hence, the communication between them requires internet connectivity. The connection can range from minimum of 3G to fast 4G (LTE).

3.2 Classes/Objects

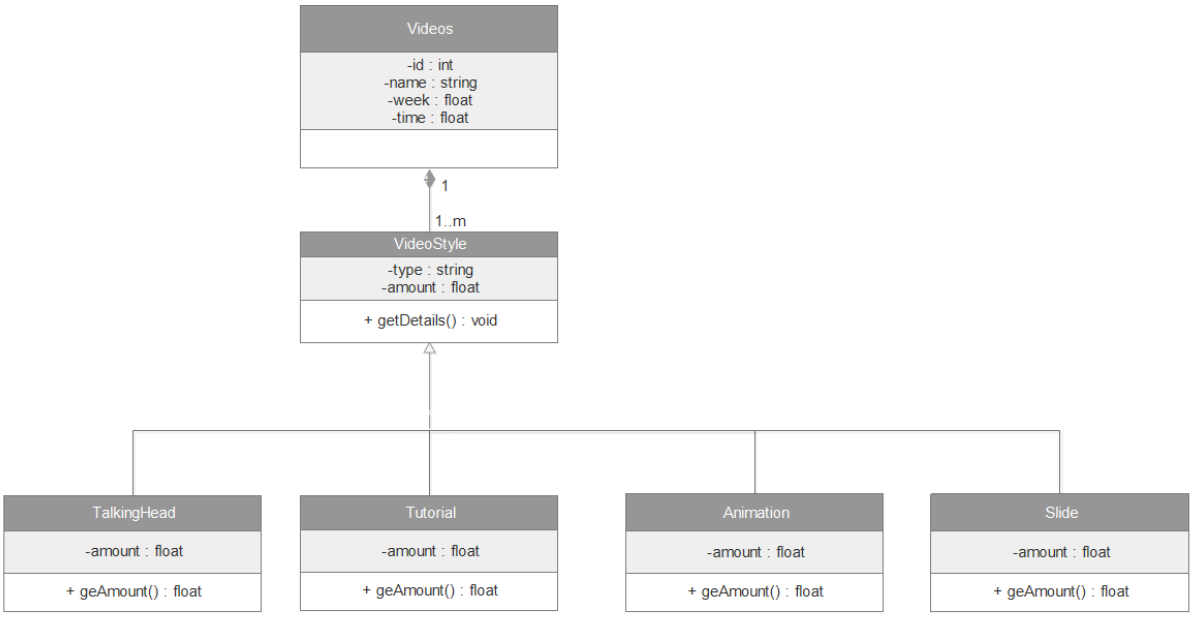

Figure 13: Class Diagram

3.3 Performance Requirements

It is expected that the video classification component will perform all the requirements stated under the product functions section. Some performance requirements identified are listed below:

- Time taken to decompose a MOOC video into image frames should be not more than 1 minute.
- Classifying the image frame into a specific video style category by the trained CNN model should be within 30 seconds given that the process is running on GPU.
- Calculating the composition level should not take more than 10 seconds.

3.4 Design Constraints

- MOOC videos only from three platforms: edX, Futurelearn and Coursera are considered.
- Training Set needs to be generated manually because, as of now, no any dataset containing the images of the different video style exists.

3.5 Software System Attributes

3.5.1 Reliability

This component should be able to perform and maintain its functions continuously without any hindrances. Given that the videos are available, it should be reliable enough with the probability of around 99% to carry out its intended operations under any circumstances.

3.5.2 Availability

As the backend process is hosted on the cloud server, the downtime may depend on the provider's services. However, the process should be available more than 95% of the time. There will be a huge system load as the new course videos are made available. However, this component is designed to carry out the tasks simultaneously such that the workload on the server remains minimal and it is up and running most of the time.

3.5.3 Security

Video Classification Component simply deals with the MOOC videos and their classification styles. The neural network model is developed using standard coding standards such that it can be easily affected by outside attacks. Encryption methods are also taken into consideration for preventing the loss of statistical information generated during the process.

3.5.4 Maintainability

The neural network is designed and developed such that it can be re-trained with new video data. It will help the model to become more robust to changes in the future.

3.6 Other Requirements

- Reusability: The specified component should be generic such that it should be suitable for use in other applications and scenarios as well.
- Interoperability: This component should be able to operate successfully by communicating with other components of the system such as: web scraping component. Similarly, it should be also efficient to exchange information with external components when needed.

4 REFERENCES

[1] P. J. Guo, J. Kim, and R. Rubin, "How video production affects student engagement," in *Proceedings of the first ACM conference on Learning @ scale conference - L@S '14*, 2014, pp. 41–50.

[2] A. Hansch, L. Hillers, K. McConachie, C. Newman, T. Schildhauer, and P. Schmidt, "Video and Online Learning: Critical Reflections and Findings from the Field," *SSRN Electron. J.*, Mar. 2015.

[3] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-Scale Video Classification with Convolutional Neural Networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[4] Lokhande B Gharde S, "A Review on Large-scale Video Classification with Recurrent Neural Network (RNN)," *Int. J. Comput. Sci. Inf. Technol.*, vol. 6, no. 2, pp. 1774–1778, 2015.

[5] Joe Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 4694–4702.

[6] M. Pratusevich, "EdVidParse: Detecting People and Content in Educational Videos," *Massachusetts Inst. Technol.*, 2013.

5 APPENDIX