Low Level Design

Face Mask Detection System

Written By : Iheb Amri

Document Version : 0.1

Contents

1. Introduction

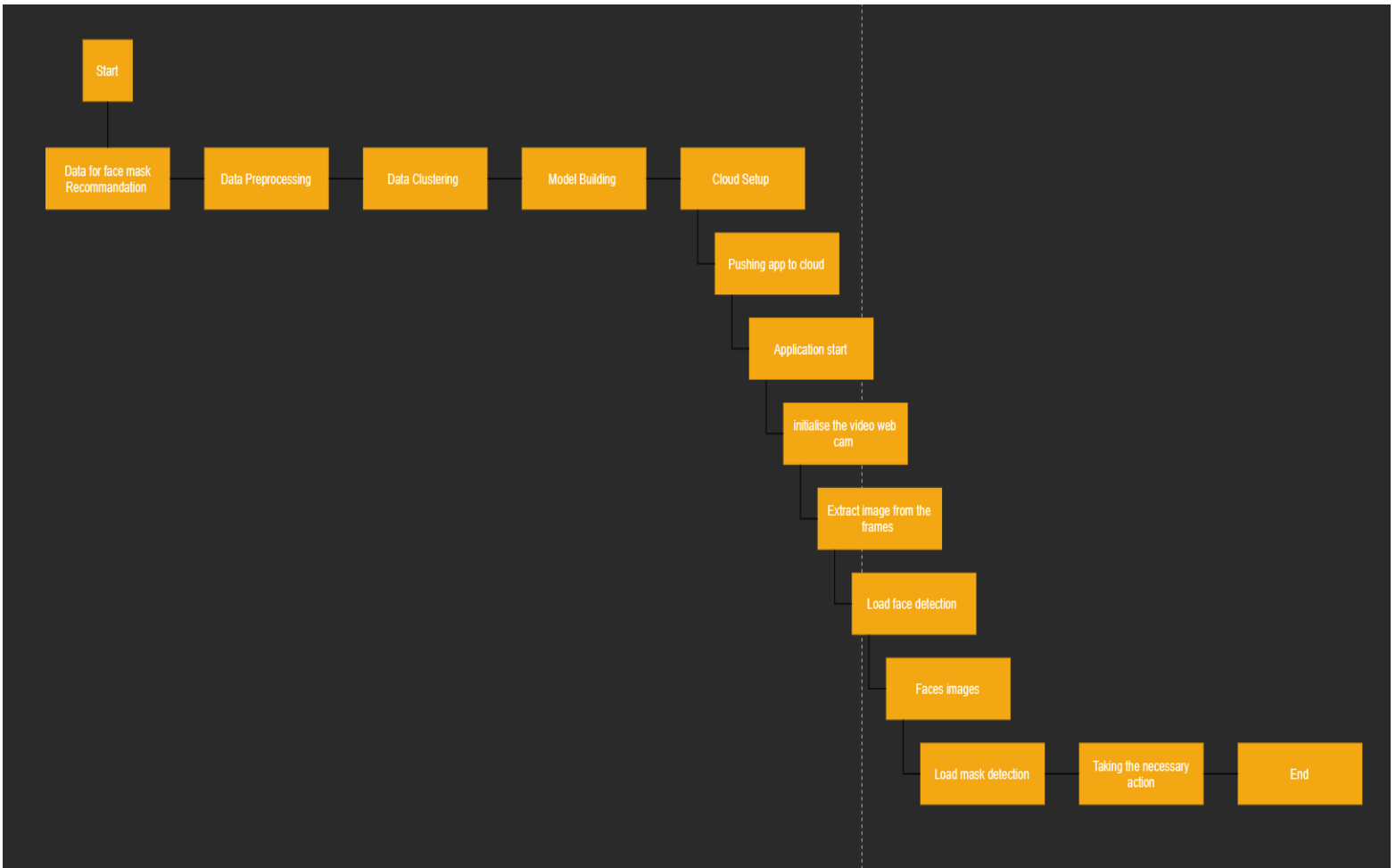
1.1. What is a Low-Level design document?

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for the Face Mask Detection System. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document.

1.2. Scope

Low-level design (LLD) is a component-level design process that follows a step-by step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work.

2. Architecture



3.1. Data Description

The development of the Face Mask Recognition model begins with collecting the data. The dataset trains data on people who use masks and who do not. The model will differentiate between people wearing masks and not.

For building the model, this study uses 1.918 data with masks and 1.915 data without a mask. At this step, the image is cropped until the only visible object is the face of the object.

The next step is to label the data. The data which has been collected is labeled into two groups; with and without a mask. After the data has been labeled, it is grouped into those two groups.

3.2. Data Preprocessing

The pre-processing phase is a phase before the training and testing of the data.

There are **four steps** in the preprocessing which are resizing image size, converting the image to the array, pre-processing input using MobileNetV2, and the last is performing hot encoding on labels.

1. The resizing image is a critical pre-processing step in computer vision due to the effectiveness of training models. The smaller size of the image, the better the model will run. In this case , the resizing of an image is making the image into 224 x 224 pixels.
2. The next step is to process all the images in the dataset into an array. The image is converted into the array for calling them by the loop function.
3. After that, the image will be used to pre-process input using MobileNetV2.
4. And the last step in this phase is performing hot encoding on labels because many machine learning algorithms cannot operate on data labeling directly. They require all input variables and output variables to be numeric, including this algorithm. The labeled data will be transformed into a numerical label, so the algorithm can understand and process the data.

3.3. Data Splitting

After the preprocessing phase, the data is split into two batches, which are training data namely 80 percent, and the rest is testing data. Each batch contains both with-mask and without-mask images.

3.4. Model Building

The next phase is building the model. There are six steps in building the model which are :

1. Constructing the training image generator for augmentation.
2. The base model with MobileNetV2.
3. Adding model parameters .
4. Compiling the model.

5. Training the model.
6. Saving the model for the future prediction process.

3.5. Testing the Model

To make sure the model can predict well, there are steps in testing the model. The first step is making predictions on the testing set.

3.6. Implementing the model.

The model implemented in the video. The video reads from frame to frame, then the face detection algorithm works. If a face is detected, it proceeds to the next process. From detected frames containing faces, reprocessing will be carried out including resizing the image size, converting to the array, and preprocessing input using MobileNetV2.

The next step is predicting input data from the saved model. Predict the input image that has been processed using a previously built model. Besides, the video frame will also be labeled that the person is wearing a mask or not along with the predictive percentage.

3.7. Deployment

We will be deploying the model onFirebase.

4. Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL is accessible to the user	1. Application URL should be defined	Application URL should be accessible to the user
Verify whether the Application loads completely for the user when the URL is accessed	1. Application URL is accessible 2. Application is deployed	The Application should load completely for the user when the URL is accessed
Verify whether the User is able to activate cameras	1. Application is accessible	The User should be able to activate cameras
Verify whether user is able to see the predictions	1. Application is accessible 2. Cameras are on	User should be able to see predictions
Verify whether user is able to see logs	1. Application is accessible 2.Cameras are on 3. The user can see predictions	User should be able to see logs
Verify whether persons without masks are captured and an email is sent to the supervisor	1. Application is accessible 2.cameras are on 3. A person without mask detected	An email should be sent with photo of the person without mask