Capstone Project Planning Document

Face Mask Detection using Convolutional Neural Networks

By

DING NICK HONG

(18039503)

Bachelor of Software Engineering (Hons)

Supervisor: Ir. Dr. MATTHEW TEOW YOK WOOI

Date: 8 July, 2021

1.0 Contents

2.0 Introduction

In this section, the project will discuss briefly the impact of the infamous Covid-19 pandemic and the challenges of training a reliable model to handle the real-life challenges of handling the quality of input to the model. Then, the project will introduce the proposed solution as well as the project's aim, objective, scope and finally the timeline in which the outcome of the system would be presented in a tabular form.

2.1 Background

Started off as cases of "viral pneumonia" in Wuhan, People's Republic of China at the end of December 2019, the infamous Covid-19 had quickly made its way to infect the rest of the world, finally forcing WHO to declare a pandemic on 11 March 2020 [1]. With the swift widespread of the virus, comes immediate countermeasures upon discovery of the symptoms and the cause of tramission. Soon, everyone was advised heavily and repeated to always wear a face mask to minimize the chances of transmitting the disease to one another. However, for varying reasons and personal choices, many have omitted or had been inconsistent of wearing a face mask in public, thus, increasing level of threat for anyone who uses public services. Therefore, creating a situation where there hadn't been more need for authorities to hold their people accountable in the public. In response to this problem, the technology world, with its attempts, had come up with many solutions for the authorities to monitor, record, and perform immediate validation of whether a person is wearing a mask or not in real-time. Many of these solutions comes from the approach of using deep-learning models to identify the person and the face mask, and Convolutional Neural Network and its varying architectures are notably trending in the sea of choices [2][3][4][5][6]. These solutions all have their own pieces of contributions and unique-ness to their models in the way they approach the problem and solve some of the specific issues of identification.

2.2 Problem Statement

Though it may be easy to assume that the project can simply input an image to a trained model and expect immediate and accurate response from the model – considering how face masks identification is less complex than other objects like hats, shoes, etc –, the real challenge here lies with the complexity of the input in terms of the background clutter (crowd), the quality of image (noises), different perspectives of the subject (view-point variation), and many more [7]. With these given circumstances, the trained model has be reliable enough to handle the environmental variables to accurately identify the face-masks, else, we would end up training multiple models for use-cases that may proof to be redundant and generally it is advisable that we should conserve our resources especially in this time of needs.

2.3 Project Aim

The aim of this project is to research on the reliability of existing models under several conditions especially low-light conditions, and come up with our very own model that merges the technology of a Face-Mask Identification model and the Low-light Image Enhancement model alongside with a web application to demonstrate the use as well as promoting the model.

2.4 Project Objectives

The objective of this project is to build increments and improve on existing models and help the authorities to monitor and understand better of the public's behaviours during the pandemic, whether they are wearing a face mask or not, regardless of the condition of the image.

- To investigate how well existing models are performing under image identificaiton challenges, especially low-light conditions.

- To design a proposed solution by integrating existing models with low-light enhancement models.
- To draw a comparison/evaluation table between the two models and record their performances under the same testing data set.
- To develop a web application to showcase the proposed solution and promote the use of the model.

2.5 Project Scope/Deliverables/Methodology

To address the some of the challenges of training a reliable model, then proceed to investigate and compare some of the existing models (if the models are available) in terms of their reliability under the influence of several conditions. The range of these challenges covers from varying image noises (Gaussian noise, Salt-and-pepper noise, Poisson noise, and more) to background clutter and view-point variation.

After that, the project will base the proposed solutions with more inclination to the problems that the project found in the existing models and create one of our own to tackle the problems, with the central focus of low-light conditions. The project will use online available dataset like Kaggle and perform a quality checking before proceeding to train the model. Various techniques will be used here to enlarge the dataset pool and variety, these techniques include data augmentation, data sequencing, and more. Throughout this process of perfecting the model, the project will also record down the technical challenges.

Finally, the model's use will be demonstrated through a web-based user interface using frontend frameworks. In order to do this though, the trained model will have to be exported to be compatible with the application's source code. Here, Tensorflow will be used to convert the model and loaded into the web application. An abstracted view of the system's architecture will also be provided to help users understand the process behind.

As a disclaimer, for what this project won’t include are the implementations of the model on real CCTVs and the web application would not have a cloud server to store all the user data, but just plainly hosted through free online hosting services like Github or Netlify. This project intends to be a proof of concept, and make the model as light as possible to be converted, imported, and integrated with other complex systems (applications, institutions, hardwares) to help solve the issue of face mask identifications.

2.6 Timeline

Research, planning, and project scope management are executed in Capstone 1. In this period, the Supervision Agreement Form (SAF), Initial Project Plan (IPP), Activity Log (AL) and Planning Document (PD) will be prepared and signed for approval by the supervisor. Literature Review, Technical Plan, Work Plan and such other details will be done here. For the exact timeline of this Capstone 1, please refer to the Gantt Chart in Activity Log.

For Capstone 2 however, the project’s focus will be directed into design, develop, and testing of the web application and the face-mask identification model. Else than that, analysis and evaluation between the existing models and the project’s proposed solution will be done. Lastly, the Final Report of this capstone project will be produced before the Presentation. For a detailed timeline of this Capstone 2, please refer to the Gantt Chart in 4.0 Work Plan section.

3.0 Literature Review

The purpose of this literature review is to understand some of the benchmarks in the current research in face mask identification models, its challenges, and some of the proposed solution to these discovered challenges. This literature review will cover information about Deep Learning, Convolutional Neural Network, image preprocessing methods, existing face mask identification models, and relate them to one another in terms of their differences, similarities, approach, results and methodology.

3.1 Convolutional Neural Network

As defined by [8], "A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other". Inspired by Visual Cortex, the architecture of a ConvNet resembles the connectivity pattern of neurons in the human brain where every individual neuron responds to stimuli/input in and only in a restricted region of the brain's visual field called the Receptive Field, and there a collection of these fields overlapping to cover the entire visual area. In the words of [9], "each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they still have a loss function (e.g. SVM/Softmax) on the last (fully-connected) layer…". Essentially, a Convolutional Neural Network in its simplest form consists typically of 3 different types of layer namely the Convolutional Layer, Pooling Layer, Fully-Connected Layer that has its own unique function to contribute to the final prediction of the model. The most typical ConvNet architecture may follow the following pattern:

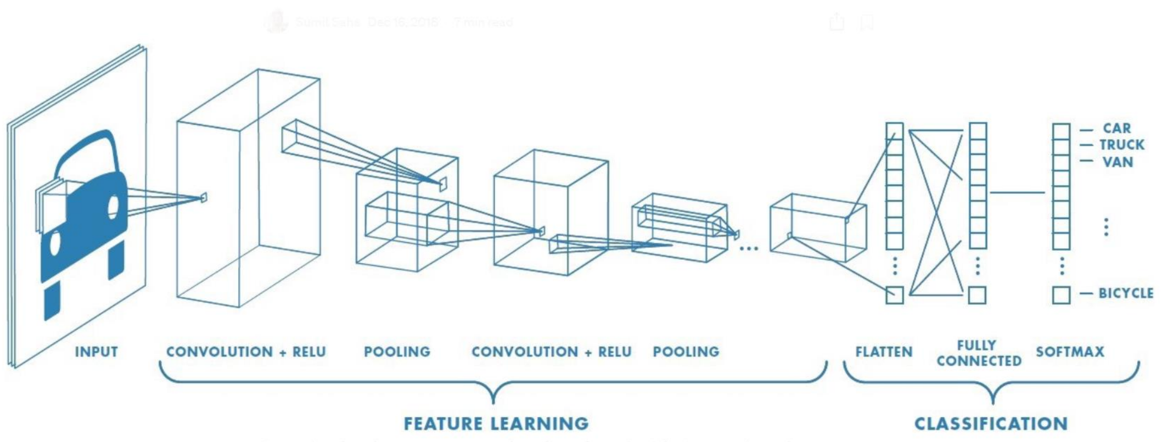


Diagram 1.0 (Convolutional Neural Network Architecture)

```
INPUT -> [[CONV -> RELU]*N -> POOL]*M -> [FC -> SOFTMAX]*K -> FC
```

With that being said, there are also other varying ConvNet architectures, each with their own unique points, but the challenges when coming up with these architectures are optimization, performance, and the underlying logics that runs through each layer, and there are a few outstanding architectures that had been published and used by many either as their choice of architecture, or as a reference. Notably, these architectures are LeNet, AlexNet, ZF Net, GoogLeNet, ResNet, and VGGNet.

- LeNet : Developed by Yann LeCun in the 1990's, it was the first successful application of Convolutional Networks. It was used to read zip codes, digits, etc.
- AlexNet : First work that popularized Convolutional Networks in Computer Vision, was similar to LeNet but the AlexNet architecture was bigger, deeper, and featured Convolutional Layers stacking on top of each other.
- ZF Net : An improvement on AlexNet by tweaking the architecture hyperparameters, notably on the expansion of size of the middle convolutional layers and making the stride and filter size on the first layer smaller.

- GoogLeNet : Its main contribution, the development of an Inception Module dramatically reduced the number of parameters in the network, it also uses Average Pooling instead of FC layers at the top of the ConvNet, optimizing the architecture by eliminating a lot of unnecessary parameters.
- VGGNet : Notably VGG-16, which consists of 16 CONV/FC layers, featuring extremely homogenous architecture that only performs 3x3 convolutions and 2x3 pooling from start to the end. Lots of their pretrained model are available for plug and play online, but the downsides are they use more memory and parameters.
- ResNet : Residual Network features skip connections and a heavy use of batch normalization, by far the state-of-the-art Convoltional Neural Network.

3.2 Image Denoising

In a real-world environment, image quality may vary drastically from their way of capture, this would result in many additional conditions added into the context of the image and therefore increasing the challenge for accurate prediction of the model since the classification model is directly influenced by the quality of the input [10]. In response to that, there exists many types of classification of found noises in images. Notably, the Gaussian noise, Salt-and-pepper noise, Poisson noise, Speckle noise, Film grain, and more.

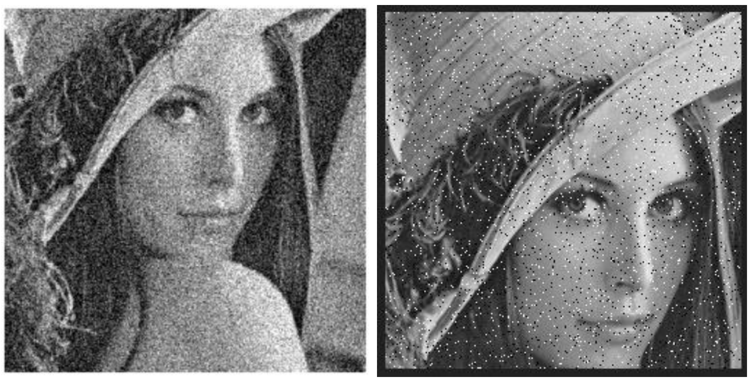


Diagram 2.0 (Gaussian Noise & Salt-and-pepper Noise)

With knowledge of these noises, the author of [10] devoted their research in the investigation of the effects of noisy images when using deep convolutional neural networks in classification tasks. The findings were impactful, the authors uses the original dataset of the trained models and degrade it by Gaussian noise and Salt& Pepper noise using their own sets of standard deviation before testing it on the trained models to find the variance in the accuracy of prediction. The authors found that the networks that are trained using data under the influence of some types of noise could be beneficial for applications that need to deal with images with varying quaity due to its more resilient nature towards noise and noise levels when compared to the models that had trained with only good quality images.

Likewise for [11], a performance analysis is drawn between conventional methods of denoising and the paper's proposed method. The paper proposed Denoising Convolutional Neural Network (DnCNN) which can effectively remove additive white Gaussian noise (AWGN) and Salt-and-pepper noise with various noise level. The conclusion was that the DnCNN produced a better PSNR value than the Histogram Equalization and Adaptive Histogram Equalization method.

On the other hand, [12] proposes a Noise and Color Bias Control module (NCBC Module) that contains a convolutional neural network and two loss functions to fix the noises and color issues in an image and even enhances low-light images, we'll save the low-light enhancement for the next section. The authors of [12] uses their NCBC Module to plug into a system that consists of many other models that performs other types of filteration. Then, they proceed to compare and analyse their enhanced images and image identification with other existing neural networks, some are even state-of-the-art architectures.

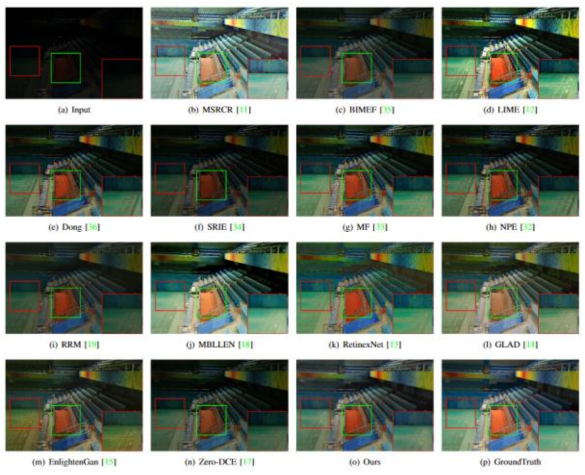


Diagram 3.0 (BLNet vs other ConvNets)

The paper concludes by proposing their own ConvNet Framework called BLNet and claims that is has fast computation speed, which most of the contributions come from their NCBC Module to fix the noises and color bias.

3.3 Low-light Enhancement

A close neighbour of image noises, low-light image is in a closely tied relationship with image noises to form one of the most challenging tasks in computer vision. For authors in [13], they set off to propose a new convolutional network called Attention U-net that works with common image file types like .PNG, .JPEG, .JPG, etc. that focuses on solving the problem of "surveillance camera security in smart city inducements without requiring the raw image file from the camera, and it can perform under the most extreme low-light conditions." [13]. On the other hand, the authors of [14] tackles the problem using a low-rank regularized Retinex model (LR3M) and demonstrates the effectiveness of their method. The differences between the previous and this paper's method is that one utilizes deep learning and training models while this one focuses on the algorithmic challenges of processing the image to enhance the low-lighting and proposes the solution in the manners of equations and calculations.

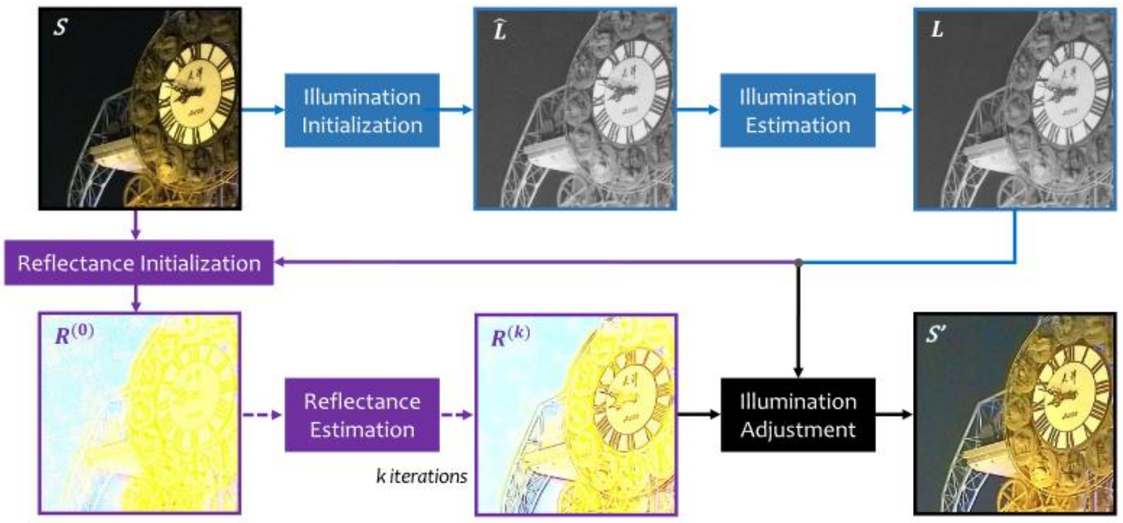


Diagram 4.0 (Low-light enhancement procedure)

The authors of [14] are the first to inject the low-rank prior into the Retinex decomposition method. Noise is considered as an explicit term in a robust Retinex model. Then, different priors are enforced to estimate illumination and reflectance maps. The reflectance map is constrained by low-rank prior, which facilitates noise removal in both the reflectance map and final enhanced result. Their method is also applicable in both images and videos.

3.4 Face Mask Identification Models

There had been multiple attempts in identifying face-mask on a person. Including [2] that allows user to open their webcam on their laptop and perform a real-time face-mask identification with very high accuracy on a personal level. However, there isn't a demonstration of other types of input like crowd's face-mask detection or other image identification challenges. On the other hand, we have [3] that also provides real-time face-mask identification but using two state-of-the-art object detection models, namely, YOLOv3 and faster R-CNN to achieve the desired outcome that the model can reliably detect a face-mask. The paper had more testing and validation with complex dataset that includes crowd detection and view-point variation, and it proofs that the model is more than capable of detecting more than one person at the same time and validate whether they are wearing a mask or not.

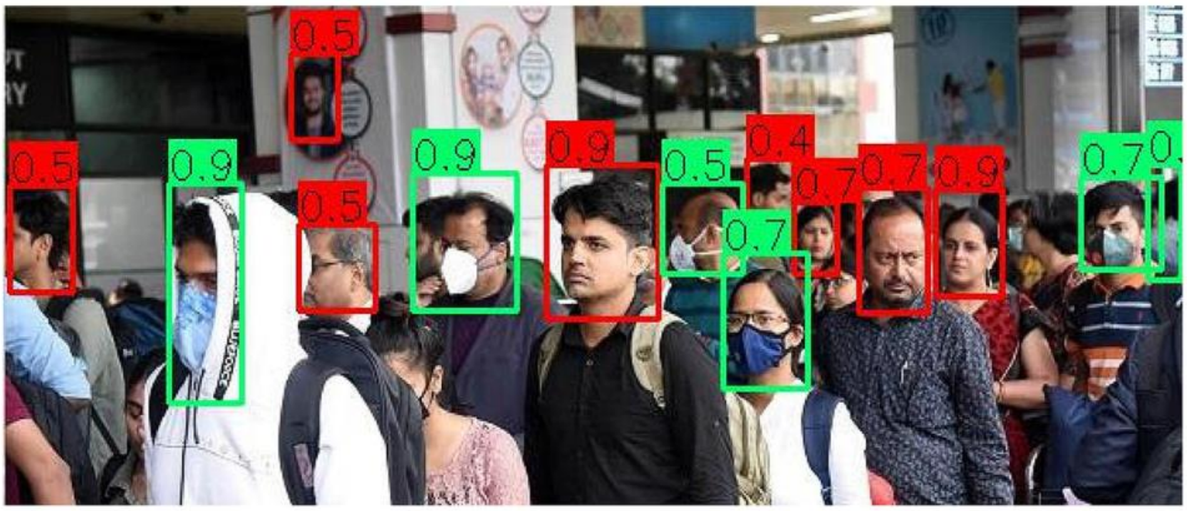


Diagram 5.0 (Face-mask detection model using YOLOv3 and faster R-CNN)

Finally, the author of paper [6] has an extra detection capability of whether the person is wearing the mask properly or not, which is a important detail given that the effects of one wearing the face-mask improperly is also as the same as not wearing a face-mask. It is good to have the variation in the detection data for authorities to have more understanding and generally a cleaner result data. There are also other face-mask identification model research done by [4] [5][15], but generally there isn't much more unique then what has already been proposed. The common trend is that no one single paper has ventured into the challenges of bad quality images like the noises and low-light conditions that we have discussed above, which makes it unreliable in night-time observation and monitoring of the crowd especially using cheaper solutions like 24/7 surveillance cameras (typically these cameras produced bad quality image and may affect detrimentally to the model's prediction).

4.0 Technical Plans

In this section, I will explain the methodology of this project in details as well as the tools that is required for the success of this project. Other than that, the functionality of the proposed solution will be included in precise details.

4.1 Proposed Solution and its Functionality

In the Litereature Review, we explored many other's works on ConvNet, Noise and Low-light enhancement, and Face Mask Identification models. We concluded that many face-mask identification solutions right now has not integrated with other technologies that allows the models to be more resilient to image noises and low-lighting conditions. Therefore, in this paper, a new combination of architectural networks will be proposed to resolve the shortcomings of existing face-mask identification models and a web application will be developed to use as a demonstration.

4.1.1 Web Application (User Interface)

Ideally, an intuitive yet simplistic web application is to be created in a UX perspective. But nowadays is hard to get away with just having full functionality of a technology in your web application, your web application will also need to look and feel premium to convince the user that they are browsing and using a legitimate application with credible background on first sight. Therefore, with all that being said, our web application shall include:

- Landing Page
    - Acts as the first page that the user will be redirected to upon accessing the webpage. Also, it has a file dropzone for users to drop their file selectively, or in a batch, otherwise, user can click on an upload button to choose their files from the file explorer. The results produced by the trained model will

also be displayed directly in the landing page, and the user can save a copy of the results.

- About/“How it works” Page
    - This page will display all the information that the user needs to know about the research, author, and sources of the technology. Moreover, a flowchart will be included as a visual representation of the webpage’s data processing procedure, so the user gets the idea of how their image files are being processed and displayed as a result.
- Github Repo Link
    - This link will be available both in the About Page and the top navigation bar the whole time for users to access the source code and improve the design of the trained model.

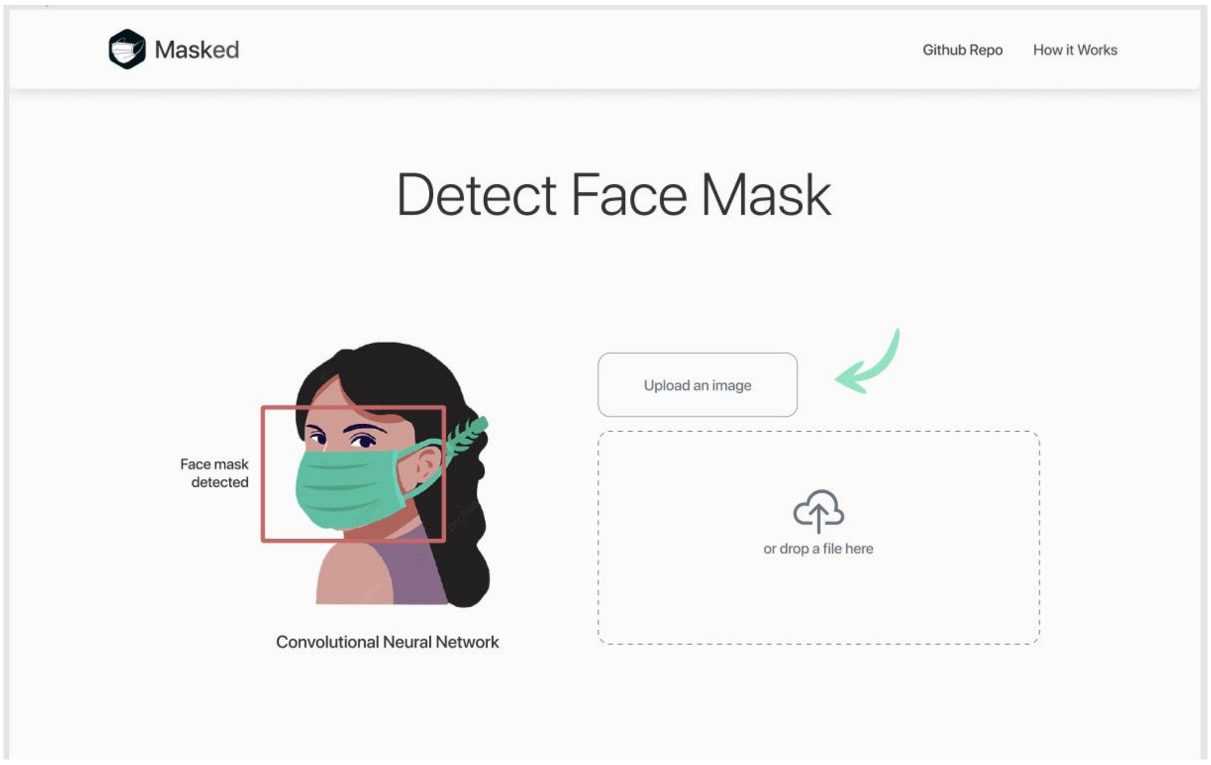


Diagram 6.0 (Masked website landing page)

And that's it, in this project we are keeping the information between the user's input until the processed results as transparent as possible, so user will have 100% confidence in using the application and stay fully in control.

4.1.2 Face-Mask Identification with Low-light Enhancement using ConvNet

For the ConvNet architecture to train the model for both the face-mask identification model and the low-light image enhancement model I am choosing the VGG-19 transfer model with the inspiration from [16] [17]. As for the dataset, I'm reusing the dataset used to train in [16] [17] to compare the between this project's improved workflow to predict the image and the previously trained models.

4.1.3 Activity Diagram and description

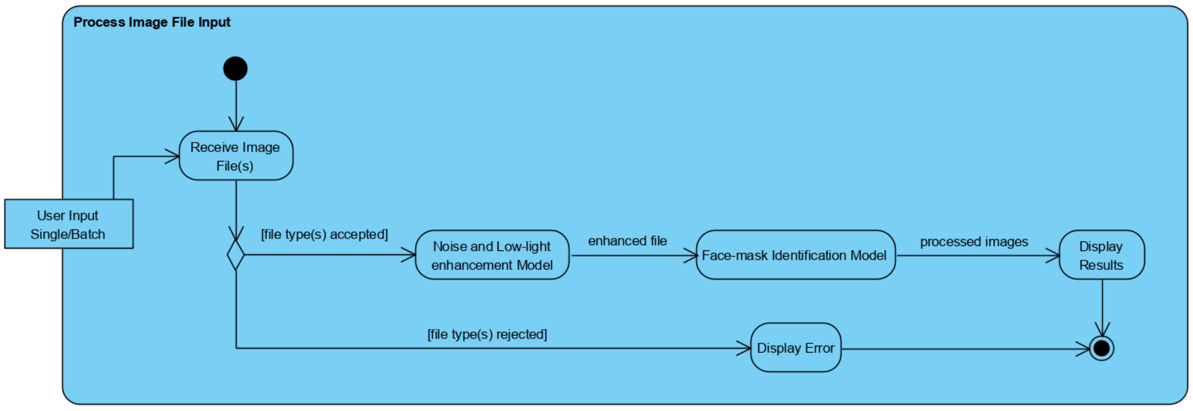


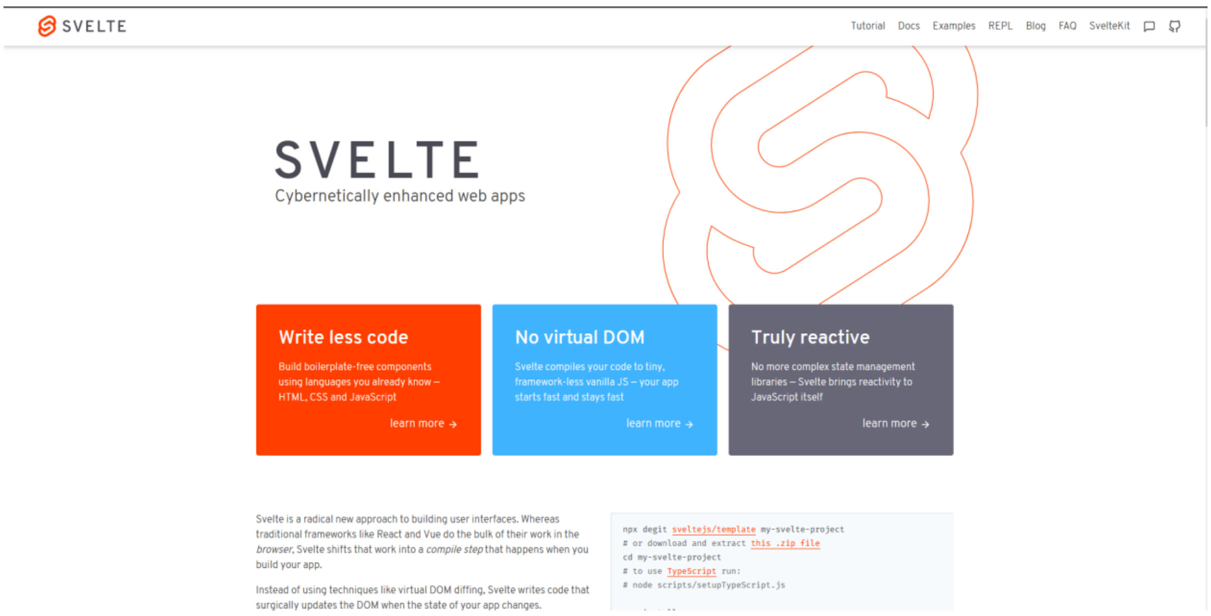
Diagram 7.0 (Activity Diagram)

As you can see clearly from the diagram, the user will only have to drop their image files as inputs, then the system will perform a validation on those files, whether it is a valid type or not (.PNG, .JPEG, etc). If not valid, show a modal to display the error message and instruct the user to input with the available file extensions. Else, perform a noise and low-light enhancement and then output into the face-mask identification model, in which the system will draw a green square (face-mask found) or red square (face-mask not found) based on the model's prediction. Finally, the image(s) are displayed in a grid to the user in the landing page, and a download button will be available for user to save a copy of the zip folder of the processed images using JSZip.

4.2 Methodologies

4.2.1 SvelteJS (svelte-file-dropzone, JSZip)

For the web development, I will use a new and popular frontend framework called SvelteJS to speed up the development process and reduce the file size.



SVELTE

Cybernetically enhanced web apps

Write less code

Build boilerplate-free components using languages you already know – HTML, CSS and JavaScript

No virtual DOM

Svelte compiles your code to tiny, framework-less vanilla JS – your app starts fast and stays fast

Truly reactive

No more complex state management libraries – Svelte brings reactivity to JavaScript itself

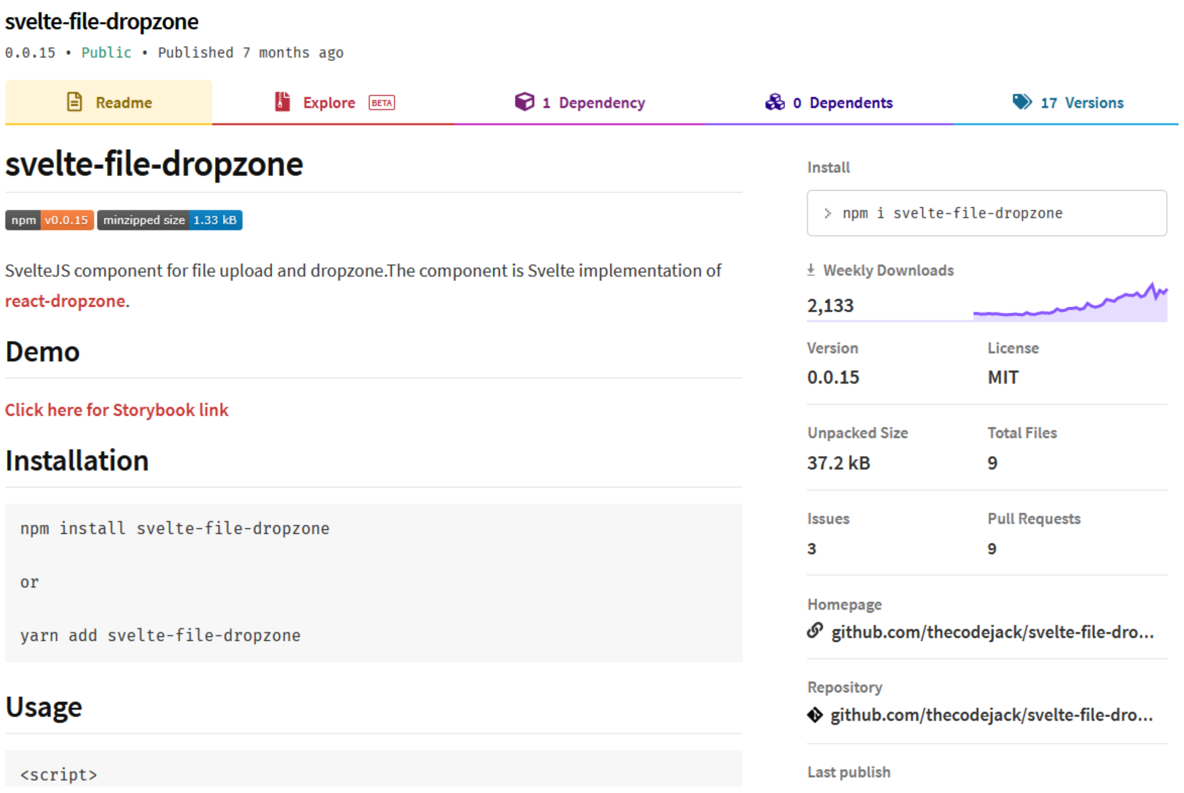
Svelte is a radical new approach to building user interfaces. Whereas traditional frameworks like React and Vue do the bulk of their work in the *browser*, Svelte shifts that work into a *compile step* that happens when you build your app.

Instead of using techniques like virtual DOM diffing, Svelte writes code that surgically updates the DOM when the state of your app changes.

```
npx degit sveltejs/template my-svelte-project
# or download and extract this .zip file
cd my-svelte-project
# to use TypeScript run:
# node scripts/setupTypeScript.js
```

Diagram 8.0 (Svelte Home Page)

Else than that, some open-source javascript library will be used to quickly achieve some of the features in the website as well. One of which is svelte-file-dropzone, this allows users to drop their files in a seamless manner and allows the developer to access and manipulate the data in an organised manner. Moreover, JSZip will be used to zip the processed images into a folder and be available for users to download a copy to their local device. Using these tools allow the developer to develop, integrate, build, test, and document quickly.


svelte-file-dropzone
0.0.15 • Public • Published 7 months ago

Readme | Explore BETA | 1 Dependency | 0 Dependents | 17 Versions

svelte-file-dropzone

npm v0.0.15 | minzipped size 1.33 kB

SvelteJS component for file upload and dropzone.The component is Svelte implementation of react-dropzone.

Demo

Click here for Storybook link

Installation

```
npm install svelte-file-dropzone

or

yarn add svelte-file-dropzone
```

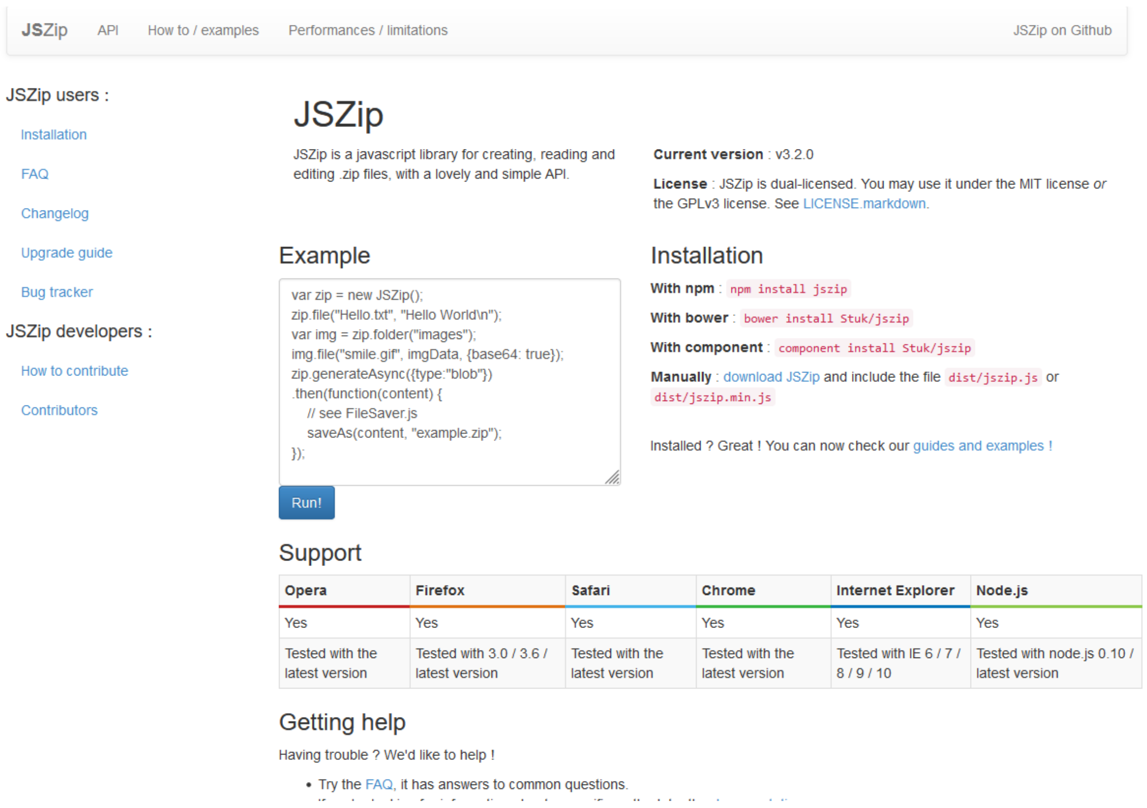
Usage

```
<script>
```

Install
```
> npm i svelte-file-dropzone
```

Weekly Downloads	2,133
Version	0.0.15
License	MIT
Unpacked Size	37.2 kB
Total Files	9
Issues	3
Pull Requests	9
Homepage	github.com/thecodejack/svelte-file-dro...
Repository	github.com/thecodejack/svelte-file-dro...
Last publish	

Diagram 9.0 (Svelte-file-dropzone)


JSZip | API | How to / examples | Performances / limitations | JSZip on Github

JSZip users :
- Installation
- FAQ
- Changelog
- Upgrade guide
- Bug tracker

JSZip developers :
- How to contribute
- Contributors

JSZip

JSZip is a javascript library for creating, reading and editing .zip files, with a lovely and simple API.

Current version : v3.2.0

License : JSZip is dual-licensed. You may use it under the MIT license *or* the GPLv3 license. See LICENSE.markdown.

Example

```
var zip = new JSZip();
zip.file("Hello.txt", "Hello World\n");
var img = zip.folder("images");
img.file("smile.gif", imgData, {base64: true});
zip.generateAsync({type:"blob"})
.then(function(content) {
    // see FileSaver.js
    saveAs(content, "example.zip");
});
```

Run!

Installation

With npm : `npm install jszip`
With bower : `bower install Stuk/jszip`
With component : `component install Stuk/jszip`
Manually : download JSZip and include the file `dist/jszip.js` or `dist/jszip.min.js`

Installed ? Great ! You can now check our guides and examples !

Support

Opera	Firefox	Safari	Chrome	Internet Explorer	Node.js
Yes	Yes	Yes	Yes	Yes	Yes
Tested with the latest version	Tested with 3.0 / 3.6 / latest version	Tested with the latest version	Tested with the latest version	Tested with IE 6 / 7 / 8 / 9 / 10	Tested with node.js 0.10 / latest version

Getting help

Having trouble ? We'd like to help !

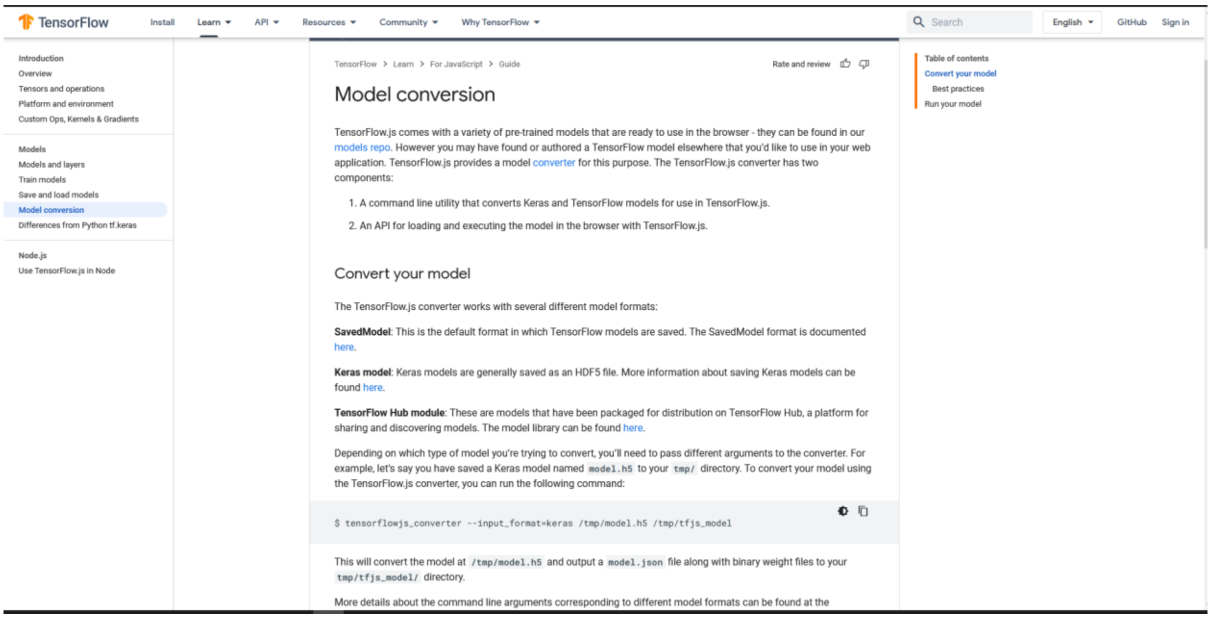
- Try the FAQ, it has answers to common questions.

Diagram 10.0 (JSZip)

These project dependecies will be downloaded using Node Package Manager (NPM), which is popular solution to include many open-source codes in the project.

4.2.2 Tensorflow

SvelteML will be download using NPM in the project repository to access Tensorflow's function as well, but I feel that Tensorflow deserves a section of its own. Because, the trained model usually gets saved in a .h5 file extension, therefore you will need to use Tensorflow's function to convert the models into a format that is small and compatible to be loaded in the frontend. So for Tensorflow, we will have two specific uses, one is to use its Model Conversion function to convert Keras Models into .BIN and .JSON file, and next, we will use SvelteML in frontend to load the models so that it works out of the box and doesn't require a specific endpoint or server to perform predictions.



Model conversion

TensorFlow.js comes with a variety of pre-trained models that are ready to use in the browser - they can be found in our models repo. However you may have found or authored a TensorFlow model elsewhere that you'd like to use in your web application. TensorFlow.js provides a model converter for this purpose. The TensorFlow.js converter has two components:

1. A command line utility that converts Keras and TensorFlow models for use in TensorFlow.js.
2. An API for loading and executing the model in the browser with TensorFlow.js.

Convert your model

The TensorFlow.js converter works with several different model formats:

SavedModel: This is the default format in which TensorFlow models are saved. The SavedModel format is documented here.

Keras model: Keras models are generally saved as an HDF5 file. More information about saving Keras models can be found here.

TensorFlow Hub module: These are models that have been packaged for distribution on TensorFlow Hub, a platform for sharing and discovering models. The model library can be found here.

Depending on which type of model you're trying to convert, you'll need to pass different arguments to the converter. For example, let's say you have saved a Keras model named `model.h5` to your `tmp/` directory. To convert your model using the TensorFlow.js converter, you can run the following command:

```
$ tensorflowjs_converter --input_format=keras /tmp/model.h5 /tmp/tfjs_model
```

This will convert the model at `/tmp/model.h5` and output a `model.json` file along with binary weight files to your `tmp/tfjs_model/` directory.

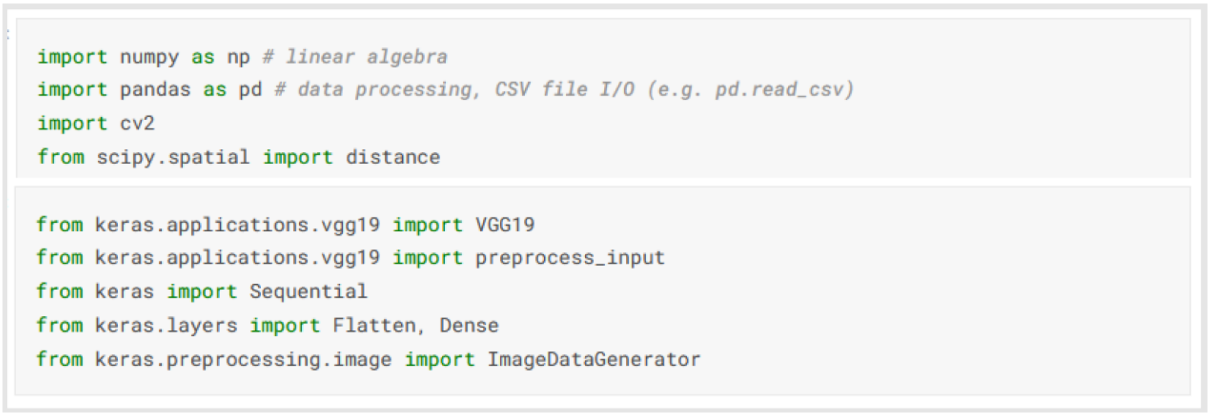
More details about the command line arguments corresponding to different model formats can be found at the

Diagram 11.0 (Tensorflow Model Conversion)

This tool is mainly used for integration with the frontend code.

4.2.3 Kaggle and Python Libraries

Many of the open-sourced, available publication of Keras mdoels and datasets are on Kaggle, which can be referenced and improved on. For this project, I am going to make use of Kaggle's online platform to copy and edit the publicated models, and many other model training tools, data processing and analytic tools will be used in synchronization to deliver the best results of the development of the models.



```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import cv2
from scipy.spatial import distance
```

```
from keras.applications.vgg19 import VGG19
from keras.applications.vgg19 import preprocess_input
from keras import Sequential
from keras.layers import Flatten, Dense
from keras.preprocessing.image import ImageDataGenerator
```

Diagram 12.0 (Sample imports of various tools from python libraries)

Here, I'll briefly explain each tools functionality and main purposes in the development of the model, meaning they will have their own contributions in training, testing, displaying, analysing, and more:

- Keras : It is a neural network library that has high-level APIs that simplifies the building and training process of model. [18]
- NumPy : Used to work with arrays, it provides an array object that works 50 more times than traditional Python lists, making it a preferred choice in Data Science and Machine Learning where data has to be processed really quickly to optimize the training of a model. [19]
- OpenCV : "OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time

operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human." [20]

- Pandas : It is a popular python-based data analysis toolkit that has diverse range of utilities including parsing mutliple file formats and converting data tables into NumPy matrix array. [21]

4.2.4 Jupyter Notebook

This tool will be used to export the Kaggle notebooks into the developer's local device and perform a local testing to ensure there are no development environment conflict. Else than that, Jupyter Notebook also allows the developer to save checkpoints so its easier for the developer to segregate and work on the parts where issues came from while preserving the state of the performed codes. For example, training a model everytime the developer adds any increment to the source code will take an unnecessarily long time, therefore, the checkpoints can help save the developer from the trouble of going through the import and training process again and again.

4.2.5 Figma

This is a design tool that allows collaborative work, but obviously Capstone 1 is an individual project. However, there are other features of Figma that is attractive to this project, like community design templates that has premade components for designing the UI/UX of the website. For example, Bootstrap is a popular CSS Framework that is used in frontend development, and Figma has a community design board for Bootstrap with all its predefined look and feel of components and typography, allowing a much faster design process of the website while maintaining consistency and aesthetic.

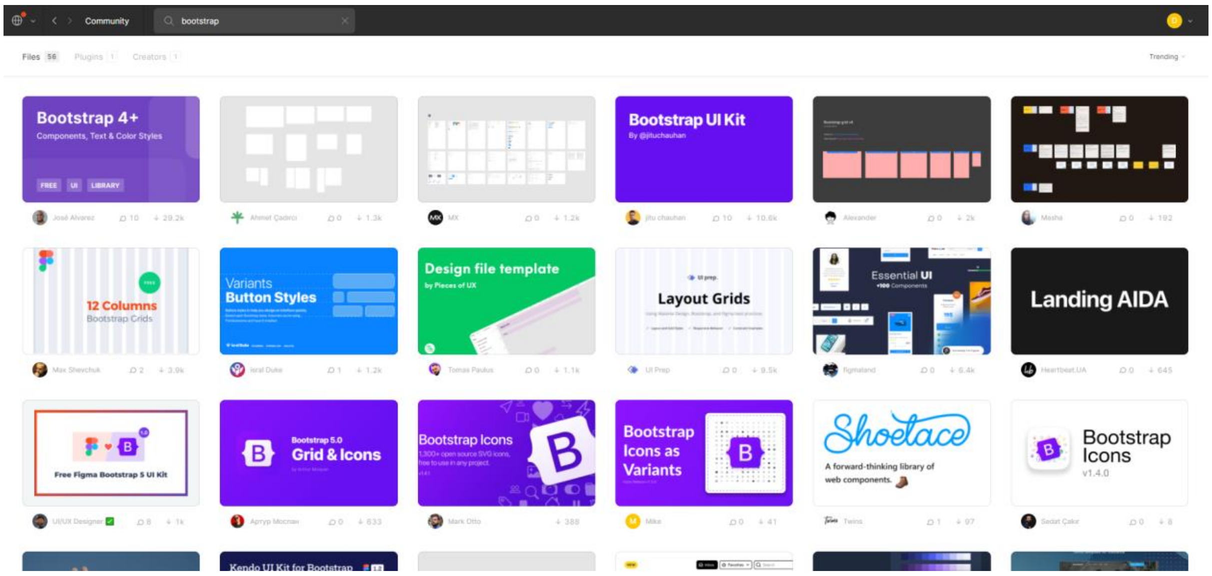


Diagram 13.0 (Figma Community Design Templates)

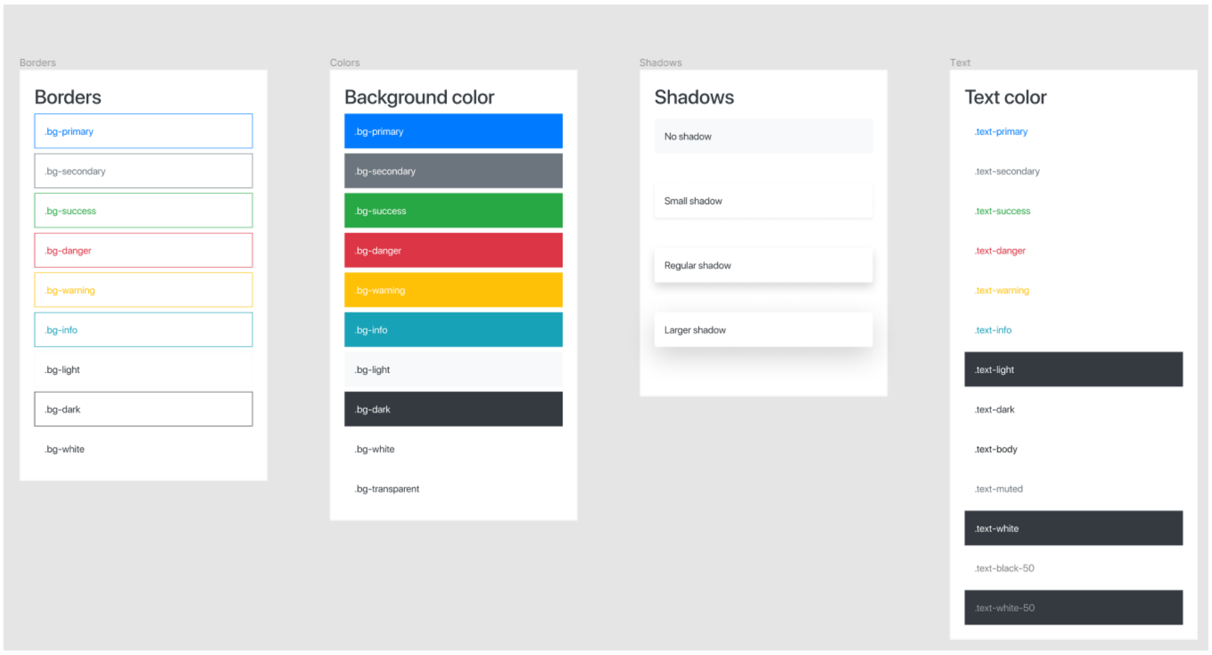


Diagram 14.0 (Bootstrap Components in Figma)

5.0 Work Plan

This section will discuss about the general work plan of the project, which includes the break down of the proposed solutions into small workable chunks that can be incremented on the system. Here, we schedule a working pace for each of the work products like development, testing, documenting, and others that will be carried out throughout the period of Capstone 2 in a Gantt Chart for ease of monitor and management.

5.1 Summary of Tasks and Milestones

1. Design the web application
    - 2 weeks
    - Predecessor : none
    - Successor : Web Development
2. Merge the models and export (Analytics)
    - 4 weeks
    - Predecessor : none
    - Successor : Web Development (Integration when loading model through Tensorflow)
3. Web development
    - 4 weeks
    - Predecessor : Web application Design
    - Successor : Testing and Debug
4. Testing and Debug
    - 4 weeks
    - Predecessor : Web Development & Models Development
    - Successor : Final Report Preparation

5. Prepare final report
    - 4 weeks
    - Predecessor : Testing and Debug
    - Successor : Presentation
6. Presentation
    - 2 weeks
    - Predecessor : Final Report Preparation
    - Successor :   none

5.2 Work activities, Risk Factors, Duration, and Acceptance Criteria

Work Activities	Work Product	Risk Factors	Acceptance Criteria	Durations
Design the web application	• Asset Gathering like icons, images, and others. • Landing Page with the dropzones, titles, and navigation. • About Page with information of technology used and more. • Mobile Responsive-ness, creating multiple that illustrates the website's UI in different sizes of screen.	• The actual development might have changes that are not reflect on the design. • Might have conflict with other imported component's design theme.	The pages are designed with consistency, reusability, and should be adaptibility to the actual development of the UI including mobile responsive-ness.	**2 weeks**
Merge the models and export (Analytics)	• Development of Baseline Face-Mask Identification Model	• The two models are not compatible with each other. • The proposed workflow will increase the processing time of each image, thus,	Develop a succesful workflow with two models integrated and working in conjunction to perform the most accurate	**4 weeks**

	• Development of Baseline Face-Mask Identification Model • Model Conversion (Tensorflow) • Create a workflow that uses both the models for prediction • Perform analysis on both models individually and in combination • Compare with other existing models' prediction	increasing the total time needed for users to get the results when uploading a bundle. • There may be redundancy when processing what may be already obvious to the face-mask identifiaction model.	prediction of whether the subject is wearing a mask or not. A clear and concise table of comparison must be drawn to illustrate the proposed solution's effectiveness as compared to the previous.	
Web development	• Load pre-trained model using SvelteML • Svelte-file-dropzone to get user input and transition to prediction using the models • JSZip to zip processed images files for user to download. • Landing and About Page with mobile responsiveness	• May be vulnerable as the models are put in frontend source code • May experience technical issues when building the file • Users using mobile may experience a harder usage when uploading files and going through the results	Must be able to load the models and implement the workflow in the frontend as seamless as possible from user input to producing the results, should be tamperproof and resilient to random user inputs	**4 weeks**

	• Navigation, routing • Web hosting	• Data loss during navigation transition due to no localstorage of processed data.		
Testing and Debug	• Usability and System Testing with some real users and gather feedback for refinement. Ease of use, accuracy of the system, and general exeprience will be recorded as a reference to refine the system before the presentation. • Debug and tweaking, for both the web development activity and models merging activity to ensure the model is perfected for mask detection. • A buffer for the developer to check off the requirements	• May discover some detrimental bugs/issues about the tools, training data, or trained model that may cause a major refactor on the system workflow and requirements. • Some user may not provide honest feedback	Must be able to quickly identify any bugs or potentially detrimental issue early on and fix it immediately upon discovery. Ensure the system has achieved every requirements gathered and is motivating for users to use the solution.	**4 weeks**

Prepare final report	• Preparation of drafts, slides, and script for presentation. • Table of contents • Compilation of work and documentation of the functionalities of the system • Compilation of analytics and comparisons between existing models	• Might include many series of changes from testing and debugging to ensure everything to up to date. • Not meeting the rubric requirements, insufficient information, or lacking analytical evidence of the proposed solution’s success	Format, layout, and references must be precise to the requirements and the work written are relevant according to the rubrics.	**4 weeks**
Presentation	• Demonstration of the web application • Guide judges through the process of training the models • Justify the selection of ConvNet architecture, tools, and frameworks • Show all analytics in a precise manner	• Not able to cover all points within the presentation time • Badly formulated delivery that confuses the judge • Failure of demonstration • Unprepared for questions. • Missing information from the actual work • Biased information delivery	Ensure the judges understand the author’s stand at least 70% and all questions from judges are accurately answered without bias, presentation should be kept within time and demonstration should be tested repeated before the actual presentation.	**2 weeks**

Gantt Chart

Activities / Tasks	Capstone Project 2													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Design the web application - Asset gathering - Landing Page - About Page - Mobile Responsiveness	■	■												
Merge the models and export (Analytics) - Development of models - Comparison of models - Workflow creation - Model conversion			■	■	■	■								
Web development - JSZip, svelte-file-dropzone, Tensorflow - Pages, navigation, mobile responsiveness - Web hosting						■	■	■	■					
Testing and Debug - Usability and System testing - Refinement - Requirements cross-checking								■	■	■	■			
Prepare final report - Drafts, presentation slides, charts - Compilation of work - Documentation of functionalities - Compilation of analytics										■	■	■	■	
Presentation - Prepare demonstartion - Rehearsal - Prepare justifications to selection of tools etc.													■	■

6.0 References

[1] OMS, “Listings of WHO’s response to COVID-19,” *World Health Organization,* 2020. https://www.who.int/news/item/29-06-2020-covidtimeline (accessed Jun. 05, 2021).

[2] M. S. Islam, E. Haque Moon, M. A. Shaikat, and M. Jahangir Alam, “A novel approach to detect face mask using CNN,” *Proc. 3rd Int. Conf. Intell. Sustain. Syst. ICISS 2020,* pp. 800–806, 2020, doi: 10.1109/ICISS49785.2020.9315927.

[3] S. Singh, U. Ahuja, M. Kumar, K. Kumar, and M. Sachdeva, “Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment,” *Multimed. Tools Appl.*, vol. 80, no. 13, pp. 19753–19768, 2021, doi: 10.1007/s11042-021-10711-8.

[4] B. Batagelj, P. Peer, V. Štruc, and S. Dobrišek, “How to Correctly Detect Face-Masks for COVID-19 from Visual Information?,” *Appl. Sci.*, vol. 11, no. 5, p. 2070, 2021, doi: 10.3390/app11052070.

[5] A. Chavda, J. Dsouza, S. Badgujar, and A. Damani, “Multi-Stage CNN Architecture for Face Mask Detection,” *2021 6th Int. Conf. Converg. Technol. I2CT 2021*, 2021, doi: 10.1109/I2CT51068.2021.9418207.

[6] M. Inamdar and N. Mehendale, “Real-Time Face Mask Identification Using Facemasknet Deep Learning Network,” *SSRN Electron. J.*, 2020, doi: 10.2139/ssrn.3663305.

[7] R. Gilani, “Main Challenges in Image Classification,” *Towards Data Science*, 2020. https://towardsdatascience.com/main-challenges-in-image-classification-ba24dc78b558.

[8] S. Saha, “A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way,”

Towards Data Science, 2018. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 (accessed Jun. 05, 2021).

[9] "CS231n Convolutional Neural Networks for Visual Recognition," *Stanford*, 2021. https://cs231n.github.io/convolutional-networks/#case (accessed Jun. 06, 2021).

[10] T. S. Nazaré, G. B. P. da Costa, W. A. Contato, and M. Ponti, "Deep convolutional neural networks and noisy images," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10657 LNCS, no. January, pp. 416–424, 2018, doi: 10.1007/978-3-319-75193-1_50.

[11] S. Thayammal, G. Sankaramalliga, S. Priyadarsini, and K. Ramalakshmi, "Performance Analysis of Image Denoising using Deep Convolutional Neural Network," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1070, p. 012085, 2021, doi: 10.1088/1757-899x/1070/1/012085.

[12] X. Wei *et al.*, "BLNet: A Fast Deep Learning Framework for Low-Light Image Enhancement with Noise Removal and Color Restoration," pp. 1–13, 2021, [Online]. Available: http://arxiv.org/abs/2106.15953.

[13] S. Ai and J. Kwon, "Extreme low-light image enhancement for surveillance cameras using attention U-net," *Sensors (Switzerland)*, vol. 20, no. 2, 2020, doi: 10.3390/s20020495.

[14] L. R. R. Model, X. Ren, W. Yang, W. Cheng, and S. Member, "LR3M : Robust Low-Light Enhancement via," vol. 29, pp. 5862–5876, 2020.

[15] M. S. Ejaz and M. R. Islam, "Masked face recognition using convolutional neural network," *2019 Int. Conf. Sustain. Technol. Ind. 4.0, STI 2019*, no. December 2019,

2019, doi: 10.1109/STI47673.2019.9068044.

[16] N. S. Chauhan, “Mask and social distancing detection using VGG19,” *Kaggle*, 2021. https://www.kaggle.com/nageshsingh/mask-and-social-distancing-detection-using-vgg19/notebook (accessed Jun. 06, 2021).

[17] Victor Basu, “Low Light Image Enhancement with CNN,” *Kaggle*, 2018. https://www.kaggle.com/basu369victor (accessed Jun. 13, 2021).

[18] “Why choose Keras?,” *Keras*. https://keras.io/why_keras/.

[19] “NumPy Introduction,” *W3schools*. https://www.w3schools.com/python/numpy/numpy_intro.asp.

[20] R. Kulhary, “OpenCV – Overview,” *Geeksforgeeks*, 2019. https://www.geeksforgeeks.org/opencv-overview/ (accessed May 27, 2021).

[21] “What is pandas in Python?,” *educative.io*. https://www.educative.io/edpresso/what-is-pandas-in-python.