Creating A Web Application to Recognize Images

498R Internship/Directed Research

By

ASSADUJJAMAN NAYEEM ID# 1620538042
ABDULLAH AL NUMAN ID#1612116042

Department of Electrical and Computer Engineering
NORTH SOUTH UNIVERSITY

Faculty Advisor

Dr. Tanzilur Rahman

Assistant Professor | Department of ECE
North South University

SUMMER 2021

Letter of Transmittal

May, 2021

To

Dr. Mohammad Rezaul Bari
Associate Professor and Chairman,
Department of Electrical and Computer Engineering,
North South University,
Bashundhara R/A, Dhaka

Through

Dr. Tanzilur Rahman
Assistant Professor,
Department of Electrical and Computer Engineering,
North South University,
Bashundhara R/A, Dhaka

Subject: Submission of 498R Internship/Directed Research Report on "Creating A Web Application to Recognize Images".

Dear Sir,

With due respect, we would like to submit Our 498R Internship/Directed Research Report on "Creating A Web Application to Recognize Images" as a part of our BSc program. The report deals with A web platform, Where User upload their images and find that what includes the image. We have collect a Tensorflow.js file which is pre-tained model and use it in our website for image recognisation purpose.

Sincerely Yours,

Assadujjaman Nayeem, Abdullah Al Numan

Department of ECE, North South University,
Bashundhara R/A, Dhaka.

Approval

The498R Internship/Directed Research Report entitled **""Creating A Web Application to Recognize Images"."** by Assadujjaman Nayeem (ID#1620538042), Abdullah Al Numan (ID# 1612116042) is approved in partial fulfillment of the requirement of the Degree of Bachelor of Science in Computer Science and Engineering on September, 2021 and has been accepted as satisfactory..

Supervisor:

....

Dr. Tanzilur Rahman

Assistant Professor
Department of Electrical and Computer Engineering
North South University
Bashundhara R/A, Dhaka.

Department Chair:

....

Dr. Mohammad Rezaul Bari

Associate Professor and Chairman
Department of Electrical and Computer Engineering
North South University
Bashundhara R/A, Dhaka.

Author's declaration

This is our truthful declaration that the **"498R Internship/Directed Research Report"** we have prepared is not a copy of any **"498R Internship/Directed Research Report"** previously made by any other team. We also express our honest confirmation in support of the fact that the said **"498R Internship/Directed Research Report"** has neither been used before to fulfill any other course related purpose nor it will be submitted to any other team or authority in future.

....

Assadujjaman Nayeem

Department of Electrical and Computer Engineering
North South University, Bangladesh.

....

Abdullah Al Numan

Department of Electrical and Computer Engineering
North South University, Bangladesh.

....

ACKNOWLEDGEMENT

First of all, we wish to express our gratitude to the Almighty for giving us the strength to perform our responsibilities and complete the report.

The 498R Internship/Directed Research program is very helpful to bridge the gap between the theoretical knowledge and real life experience as part of Bachelor of Science (BSc) program. This report has been designed to have a practical experience through the theoretical understanding. Furthermore, **North South University** gave us plenty of privileges regarding technical facilities. So, we also like to admire the cooperation of our authority.

We also acknowledge our profound sense of gratitude to all the teachers who have been instrumental for providing us the technical knowledge and moral support to complete the project with full understanding.

We would like to convey our gratitude to our faculty **Dr. Tanzilur Rahman** for his stimulating inspiration, kind guidance, valuable suggestions, sagacious advice and kind cooperation throughout the period of word undertaken, which has been instrumented in the success of our project. At this level of understanding it is often difficult to understand the wide spectrum of knowledge without proper guidance and advice. His suggestions and guidance have made the report a good manner.

We like to express our heartiest gratitude to our family and friends for their moral support to carve out this project.

Abstract

Computer Vision is the branch of the science of computers and software systems which can recognise as well as understand images and scenes. Computer Vision is consists of various aspects such as image recognition, object detection, image generation, image super-resolution and many more. Object detection is widely used for face detection, vehicle detection, pedestrian counting, web images, security systems and self-driving cars. In this project, we are using highly accurate object detection-algorithms and methods. Using these methods and algorithms, based on deep learning which is also based on machine learning require lots of mathematical and deep learning frameworks understanding by using dependencies such as TensorFlow, we can detect each and every object. This also includes the accuracy of each method for identifying objects. Some other aspects of image recognition include image restoration, object recognition, and scene reconstruction.These capabilities are typically embedded inside intelligent applications. Image recognition software can be used by data scientists to train image recognition models, as well as developers looking to add image recognition features to other software.

TABLE OF CONTENTS

List of Tables

Table **Page**

LIST OF FIGURES

FIGURE **Page**

CHAPTER 1

INTRODUCTION

A few years ago, the creation of the software and hardware image processing systems was mainly limited to the development of the user interface, which most of the programmers of each firm were engaged in. The situation has been significantly changed with the advent of the Windows operating system when the majority of the developers switched to solving the problems of image processing itself. However, this has not yet led to the cardinal progress in solving typical tasks of recognising faces, car numbers, road signs, analysing remote and medical images, etc. Each of these "eternal" problems is solved by trial and error by the efforts of numerous groups of the engineers and scientists. As modern technical solutions are turn out to be excessively expensive, the task of automating the creation of the software tools for solving intellectual problems is formulated and intensively solved abroad. In the field of image processing, the required tool kit should be supporting the analysis and recognition of images of previously unknown content and ensure the effective development of applications by ordinary programmers. Just as the Windows toolkit supports the creation of interfaces for solving various applied problems. Object recognition is to describe a collection of related computer vision tasks that involve activities like identifying objects in digital photographs. Image classification involves activities such as predicting the class of one object in an image. Object localisation is refers to identifying the location of one or more objects in an image and drawing an abounding box around their extent. Object detection does the work of combines these two tasks and localises and classifies one or more objects in an image. When a user or practitioner refers to the term “object recognition“, they often mean “object detection“. It may be challenging for beginners to distinguish between different related computer vision tasks. So, we can distinguish between these three computer vision tasks with this example: Image Classification: This is done by Predict the type or class of an object in an image. Image classification also involves assigning a class label to an image, whereas object localisation involves drawing a bounding box around one or more objects in an image. Object detection is always more challenging and combines these two tasks and draws a bounding box around each object of interest in the image and assigns them a class label. Together, all these problems are referred to as object recognition. Object recognition refers to a collection of related tasks for identifying objects in digital photographs. Region-based Convolutional Neural Networks is a family of techniques for addressing object localisation and recognition tasks, designed for model performance.

1.1 Engineering Problem

There are lots of Image processing model out there. But a common people can not use it until it is user friendly. In that purpose we have built a website for user and deploy the image processing model in our website so that user can use their image to see the magic of machine learning.

1.2 Motivation

Image processing can be used in Powering self-driving cars, for a self drive car it needs to decide what in-front of the car based on the object the car can decide what should be done. Medicine – the use of photo recognition for fast and accurate medical diagnoses based on data obtained from medical photos. According to specialists, this type of technology may be the first in the area of recognizing cancerous lesions, for example: melanoma. It works in perfect alignment with augmented reality, and in fact is part of the capabilities inherent to AR. Recognizing real-life objects and augmenting them with valuable information, hints and interactivity can simplify our lives while also enhancing both industrial and academic training. there are lots of field of using this image processing. That is why we have immense attraction to build web application for user and Machine Learning engineer so that it can be easy to integrate those machine learning model and give a good user interface to the user and they can easily use those model and take advantage of technological world.

1.3 498R Internship/Directed Research Outline

The entire 498R internship/Directed Research project is described within four chapters. Each section covered about a certain task of the project which is summarized below,

(I) Chapter one gives an introduction of the our project and tells us why we have done this project.

(II) Chapter two covers an extensive literature review of previous works on the related work that tells us several website are giving that idea to implement a web application and integrate the image processing model.

(III) Chapter three is about the technical description of our idea. In this section, we are trying to describe the problem we are solving. And we also tried to describe our uniqueness in that project. It also describe the methodology of the project.That means it demonstrates the outlook of our project and gives the details the working process of our website.

(IV) Chapter five discuss the complications and limitations of our research project.

CHAPTER 2

Related Works

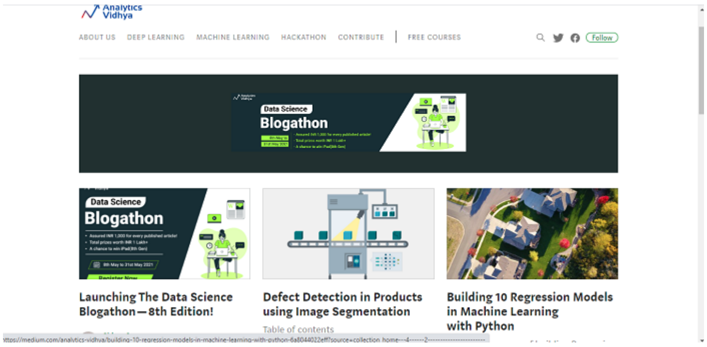
There are lots of Image processing project have done through out the computer science field like, skin cancer detection, image recognition, human pose detection, age detection and a lot of things. For developing our web application we had to study some existing related work of image recognition, so that we can detect our image.

2.1 Analytic Vidhya

Analytics vidhya[1] is a website where image processing model can be found user can take those model and integrate these model in their website. in that case user do not need to have any machine learning code knowledge. In Analytics vidya website lots of machine learning engineering guid the user how to develope a user interface and how they can integrate those model to the website.

2.2 Google Lens

This fantastic app[3] allows capturing images with a smartphone camera and then performing an image-based search on the web. It works just like Google Images reverse search by offering users links to pages, Wikipedia articles, and other relevant resources connected to the image.



Launching The Data Science Blogathon—8th Edition!

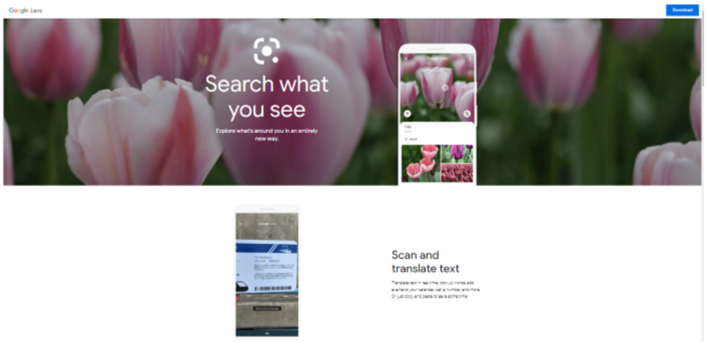
Defect Detection in Products using Image Segmentation

Table of contents

Building 10 Regression Models in Machine Learning with Python

https://medium.com/analytics-vidhya/building-10-regression-models-in-machine-learning-with-python-6a8044022eff?source=collection_home---4------2-----------------------

FIGURE 2.1. This is the home page of Analitic Vidya



Search what you see

Explore what's around you in an entirely new way.

Scan and translate text

FIGURE 2.2. Home page of google lens

2.3 Computer vision Fruit grading Image processing

This is an article[7] where the researcher have created an image processing model which can give the grade of fruit. That means user can upload their fruits image and they will see the result of the freshness of that fruit. In this article we have found that there is a model they have made ready but they did not made any website where user can use this model.

CHAPTER 3

METHODOLOGY

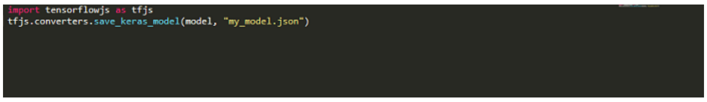
On many occasions, we always like explaining the results of our machine learning models to our website or to people who are not familiar with these topics. As an example, suppose we have a Deep Learning model that classify images, such as dogs, cats, . . . This model has good results in a test set or with some images that we have tried, but we would like that others people use this model and see the result, In that purpose it is better if they use it from a web application. This is what this implementation methodology is all about. We will try to show creating process of a web application using the tensorflow.js library and the MobileNet model. It is only necessary to know the basics in HTML and JavaScript. After following this tutorial, we have an application similar to the following:

3.1 Tensolflow.js

Tensorflow.js[6] is a library developed by Google where you can generate, from Javascript, Machine Learning models, specifically Deep Learning. Advantages of using tensorflow.js:.

3.1.1 Import Model

Import an existing previously trained model and use it in our browser. The existing model can be created with Tensorflow or Keras. For this we have to install the tensorflowjs



```
import tensorflowjs as tfjs
tfjs.converters.save_keras_model(model, "my_model.json")
```

FIGURE 3.1. Import model using tensorflow.js

library and run the following script we have taken information to deploy it in our evironment from Importing a Keras model into TensorFlow.js[4] topic.

3.1.2 Pre-Trained Model

For this application we are going to use a trained model[5], with ImageNet images, called MobileNet. MobileNet is an architecture of Convolutionary Neural Networks (CNN) which was created to be executed on mobile devices[1]. As we know, a Convolutionary Neural Network is a Deep Learning algorithm used to learn Machine Learning models related to images, such as: classifying images, detecting objects in an image or video,... [2].

3.2 Web Applicatin

After having pre-trained model[5] We will make our web application to recognize images. For this we only need a text editor, programmer can use notepad ++, sublime or a simple block of notes (with which programmer feel more comfortable). We have used sublime.text.

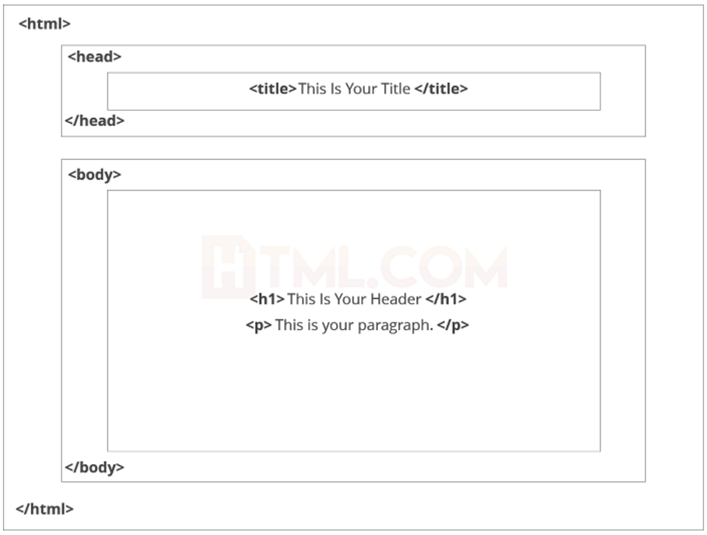
3.2.1 Sublime Text

Sublime Text is a text editor written in C++ and Python available on windows, Mac and Linux. A text editor is a program developers write their code in. History behind it goes back to 2007 when its creator Jon Skinner quit his job at Google in order to pursue one of his dreams : Create a better text editor

3.2.2 HTML

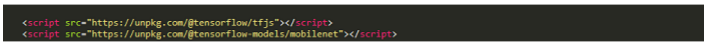
We have started with HTML, this will be responsible for structuring the web page. The file will be called index.html:

(I) In order for us to use the tensorflow.js library we need to call it from HTML as follows:



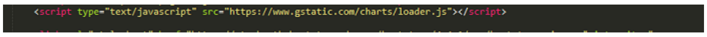
```
<html>
<head>
<title>This Is Your Title </title>
</head>
<body>
<h1>This Is Your Header </h1>
<p> This is your paragraph. </p>
</body>
</html>
```

FIGURE 3.2. Basic structure of HTML



```
<script src="https://unpkg.com/@tensorflow/tfjs"></script>
<script src="https://unpkg.com/@tensorflow-models/mobilenet"></script>
```

FIGURE 3.3. Import tensorflow.js in Html file



```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
```

FIGURE 3.4. Importing code of probabilities and decisions of predictions

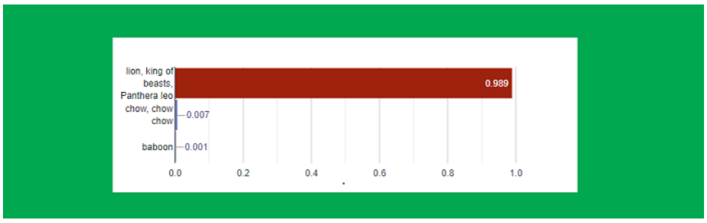


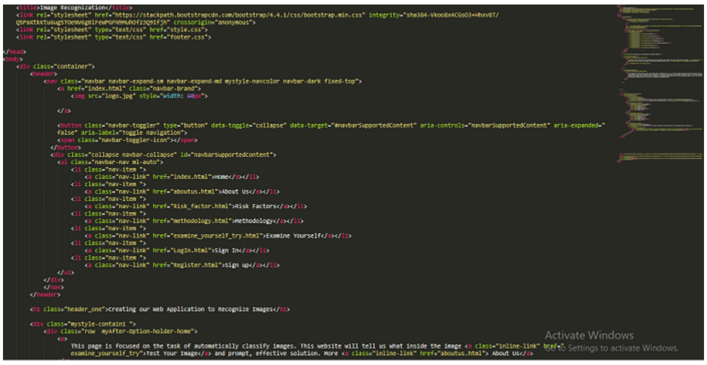
FIGURE 3.5. Out Put of the prediction

(II) After that We will show the probabilities and decisions of the predictions of the MobileNet model:

(III) The output as follows:

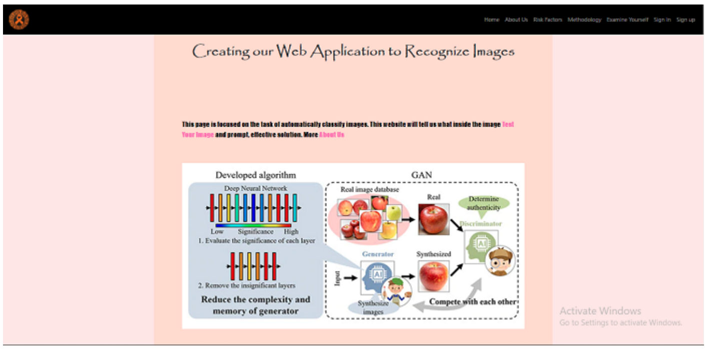
(IV) We will add some text on Home page and home page will look like:

(V) the output is as follows:



```
    <title>Image Recognization</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/
    Q6PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="style.css">
    <link rel="stylesheet" type="text/css" href="footer.css">

</head>
<body>
    <div class="container">
        <header>
            <nav class="navbar navbar-expand-sm navbar-expand-md mystyle-navcolor navbar-dark fixed-top">
                <a href="index.html" class="navbar-brand">
                    <img src="logo.jpg" style="width: 60px">

                </a>

                <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="
                false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
               </button>
               <div class="collapse navbar-collapse" id="navbarSupportedContent">
                <ul class="navbar-nav ml-auto">
                    <li class="nav-item ">
                        <a class="nav-link" href="index.html">Home</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="aboutus.html">About Us</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="Risk_factor.html">Risk Factors</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="methodology.html">Methodology</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="examine_yourself_try.html">Examine Yourself</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="LogIn.html">Sign In</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="Register.html">Sign up</a></li>
                </ul>
            </div>
            </nav>
        </header>

        <h1 class="header_one">Creating our Web Application to Recognize Images</h1>

        <div class="mystyle-containi ">
            <div class="row  myAfter-option-holder-home">
                <p>
                    This page is focused on the task of automatically classify images. This website will tell us what inside the image <a class="inline-link" href="
                    examine_yourself_try">Test Your Image</a> and prompt, effective solution. More <a class="inline-link" href="aboutus.html"> About Us</a>
```

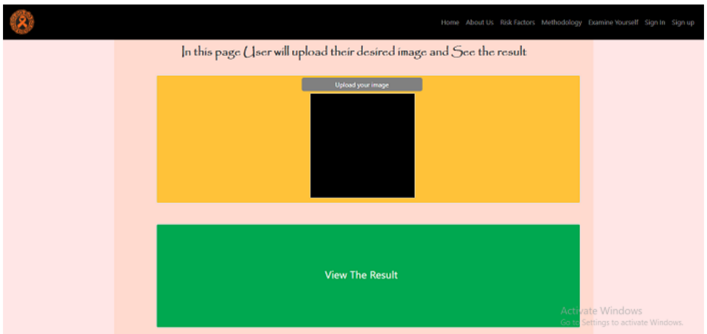
FIGURE 3.6. Home page text code of our web application



Creating our Web Application to Recognize Images

This page is focused on the task of automatically classify images. This website will tell us what inside the image Test Your Image and prompt, effective solution. More About Us

FIGURE 3.7. out put of the home page text code

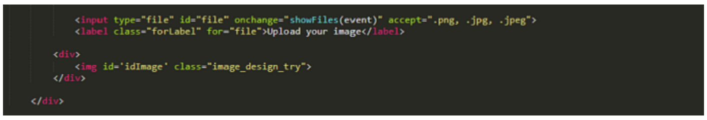


In this page User will upload their desired image and See the result

FIGURE 3.8. Code of preview image in the web application

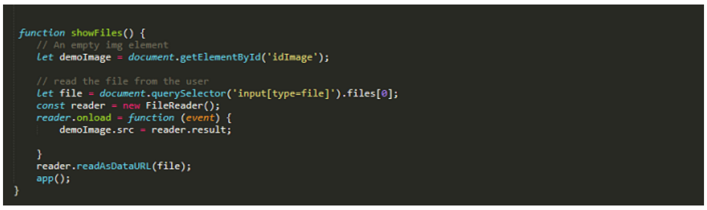
(VI) Now we are going to add a section to upload images in our web application: The MobileNet model has been trained with images of size 224 * 224, so we will transform the images at these scales. Result of upload image out put file looks like

(VII) The JavaScript function that we will use to upload an image and display it, is called showFiles(event):



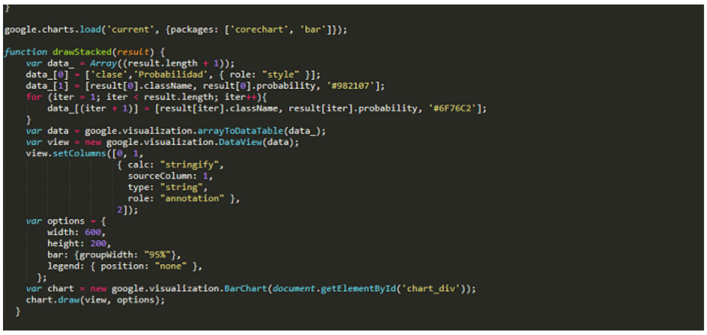
```
            <input type="file" id="file" onchange="showFiles(event)" accept=".png, .jpg, .jpeg">
            <label class="forLabel" for="file">Upload your image</label>

        <div>
            <img id='idImage' class="image_design_try">
        </div>

    </div>
```

FIGURE 3.9. Code section of show file function



```
function showFiles() {
    // An empty img element
    let demoImage = document.getElementById('idImage');

    // read the file from the user
    let file = document.querySelector('input[type=file]').files[0];
    const reader = new FileReader();
    reader.onload = function (event) {
        demoImage.src = reader.result;

    }
    reader.readAsDataURL(file);
    app();
}
```

FIGURE 3.10. Uploading image and peview it to the web application code



```
}

google.charts.load('current', {packages: ['corechart', 'bar']});

function drawStacked(result) {
    var data_ = Array((result.length + 1));
    data_[0] = ['clase','Probabilidad', { role: "style" }];
    data_[1] = [result[0].className, result[0].probability, '#982107'];
    for (iter = 1; iter < result.length; iter++){
        data_[(iter + 1)] = [result[iter].className, result[iter].probability, '#6F76C2'];
    }
    var data = google.visualization.arrayToDataTable(data_);
    var view = new google.visualization.DataView(data);
    view.setColumns([0, 1,
                    { calc: "stringify",
                      sourceColumn: 1,
                      type: "string",
                      role: "annotation" },
                    2]);
    var options = {
        width: 600,
        height: 200,
        bar: {groupWidth: "95%"},
        legend: { position: "none" },
      };
    var chart = new google.visualization.BarChart(document.getElementById('chart_div'));
    chart.draw(view, options);
  }
```

FIGURE 3.11. Following code will give the google chart of prediction

3.2.3 JavaScript

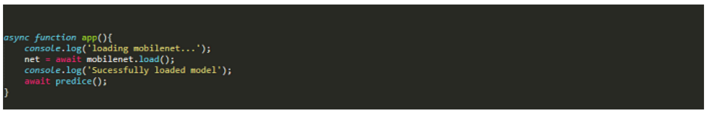
JavaScript can be abbreviated as JS[**?**]. It is a high-level object oriented programming language. It also uses for advanced design issue. It can be used to, handle requests and responses and also add dynamic behavior and also store information on a website. JavaScript will be responsible for uploading images, calling the tensorflow.js function and making predictions using the MobileNet model. The file will be calledindex.js:

(I) Frist we creat a function to upload images and display them in the browser:

(II) The following script will graph the probabilities of the predictions on a bar chart. We will use the library Google:

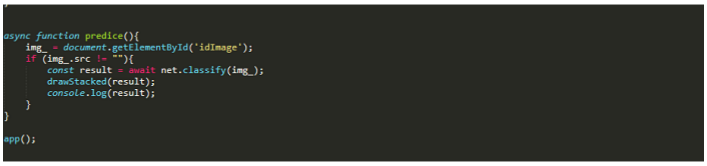
(III) Now we are going to load the pre-trained MobileNet model:

(IV) Finally, we predict the upload image.



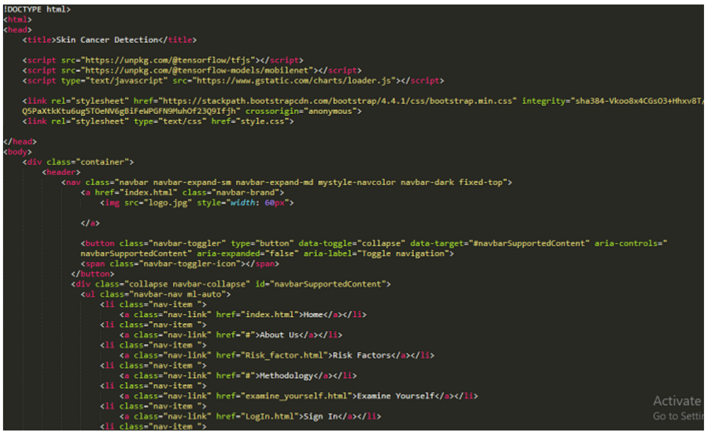
```
async function app(){
    console.log('loading mobilenet...');
    net = await mobilenet.load();
    console.log('Sucessfully loaded model');
    await predice();
}
```

FIGURE 3.12. Pre-Trained model code



```
async function predice(){
    img_ = document.getElementById('idImage');
    if (img_.src != ""){
        const result = await net.classify(img_);
        drawStacked(result);
        console.log(result);
    }
}

app();
```

FIGURE 3.13. Prediction code of target image



```
!DOCTYPE html>
<html>
<head>
    <title>Skin Cancer Detection</title>

    <script src="https://unpkg.com/@tensorflow/tfjs"></script>
    <script src="https://unpkg.com/@tensorflow-models/mobilenet"></script>
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>

    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGsO3+Hhxv8T
    Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="style.css">

</head>
<body>
    <div class="container">
        <header>
            <nav class="navbar navbar-expand-sm navbar-expand-md mystyle-navcolor navbar-dark fixed-top">
                <a href="index.html" class="navbar-brand">
                    <img src="logo.jpg" style="width: 60px">

                </a>

                <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="
                navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
               </button>
               <div class="collapse navbar-collapse" id="navbarSupportedContent">
                <ul class="navbar-nav ml-auto">
                    <li class="nav-item ">
                        <a class="nav-link" href="index.html">Home</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="#">About Us</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="Risk_factor.html">Risk Factors</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="#">Methodology</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="examine_yourself.html">Examine Yourself</a></li>
                    <li class="nav-item ">
                        <a class="nav-link" href="LogIn.html">Sign In</a></li>
                    <li class="nav-item ">
```

FIGURE 3.14. This code is the complete html code to image preview

3.2.4 Putting all codes together

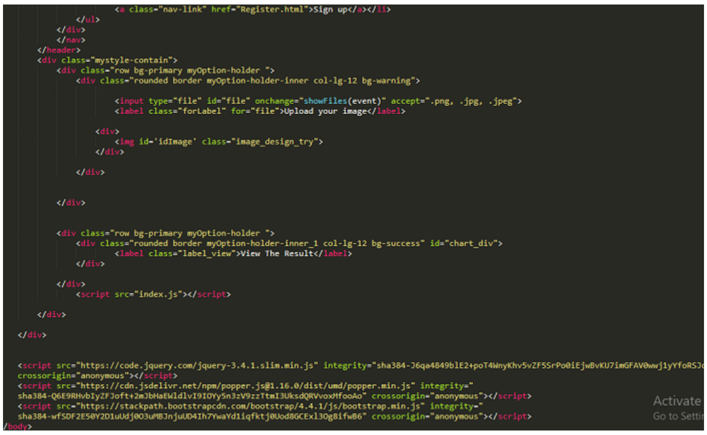
In this Section We have putted all codes together for test images page

3.2.4.1 HTML (File index.html).

Putting all codes together of Index.html page and the image is given below.

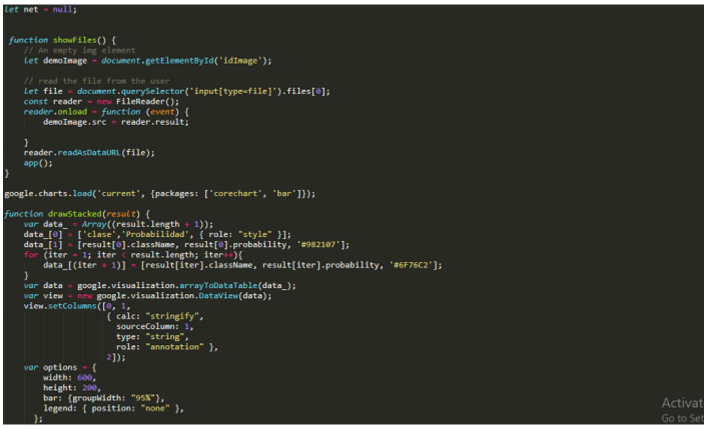
3.2.4.2 JavaScript (File index.js)

Putting all JavaScript code together and the screenshot of those code is given below.



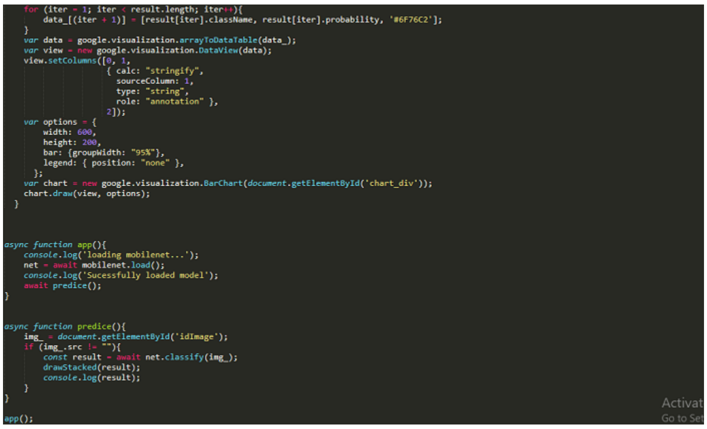
```
                    <a class="nav-link" href="Register.html">Sign up</a></li>
                </ul>
            </div>
            </nav>
        </header>
        <div class="mystyle-contain">
            <div class="row bg-primary myOption-holder ">
                <div class="rounded border myOption-holder-inner col-lg-12 bg-warning">

                    <input type="file" id="file" onchange="showFiles(event)" accept=".png, .jpg, .jpeg">
                    <label class="forLabel" for="file">Upload your image</label>

                <div>
                    <img id='idImage' class="image_design_try">
                </div>

                </div>


            </div>


            <div class="row bg-primary myOption-holder ">
                <div class="rounded border myOption-holder-inner_1 col-lg-12 bg-success" id="chart_div">
                    <label class="label_view">View The Result</label>
                </div>

            </div>
                <script src="index.js"></script>

        </div>

    </div>


    <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js" integrity="sha384-J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoRSJc
    crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity="
    sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo" crossorigin="anonymous"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js" integrity="
    sha384-wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6" crossorigin="anonymous"></script>
/body>
```

FIGURE 3.15. This code is the complete html code to image preview



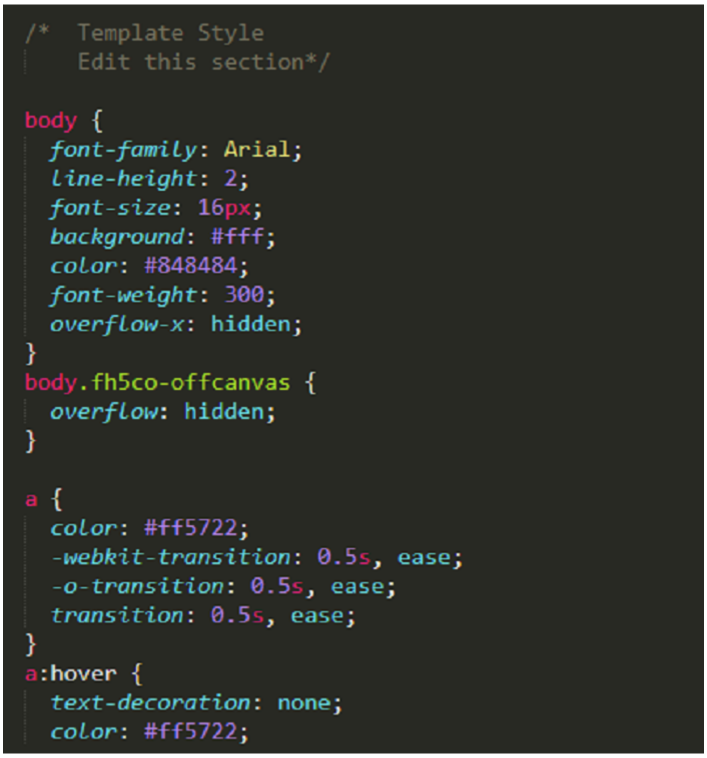
```
let net = null;


 function showFiles() {
    // An empty img element
    let demoImage = document.getElementById('idImage');

    // read the file from the user
    let file = document.querySelector('input[type=file]').files[0];
    const reader = new FileReader();
    reader.onload = function (event) {
        demoImage.src = reader.result;

    }
    reader.readAsDataURL(file);
    app();
}

google.charts.load('current', {packages: ['corechart', 'bar']});

function drawStacked(result) {
    var data_ = Array((result.length + 1));
    data_[0] = ['clase','Probabilidad', { role: "style" }];
    data_[1] = [result[0].className, result[0].probability, '#982107'];
    for (iter = 1; iter < result.length; iter++){
        data_[(iter + 1)] = [result[iter].className, result[iter].probability, '#6F76C2'];
    }
    var data = google.visualization.arrayToDataTable(data_);
    var view = new google.visualization.DataView(data);
    view.setColumns([0, 1,
                     { calc: "stringify",
                       sourceColumn: 1,
                       type: "string",
                       role: "annotation" },
                     2]);
    var options = {
        width: 600,
        height: 200,
        bar: {groupWidth: "95%"},
        legend: { position: "none" },
      };
```

FIGURE 3.16. his code is the total JavaScript code for prediction



```
    for (iter = 1; iter < result.length; iter++){
        data_[(iter + 1)] = [result[iter].className, result[iter].probability, '#6F76C2'];
    }
    var data = google.visualization.arrayToDataTable(data_);
    var view = new google.visualization.DataView(data);
    view.setColumns([0, 1,
                     { calc: "stringify",
                       sourceColumn: 1,
                       type: "string",
                       role: "annotation" },
                     2]);
    var options = {
        width: 600,
        height: 200,
        bar: {groupWidth: "95%"},
        legend: { position: "none" },
      };
    var chart = new google.visualization.BarChart(document.getElementById('chart_div'));
    chart.draw(view, options);
  }



async function app(){
    console.log('loading mobilenet...');
    net = await mobilenet.load();
    console.log('Sucessfully loaded model');
    await predice();
}


async function predice(){
    img_ = document.getElementById('idImage');
    if (img_.src != ""){
        const result = await net.classify(img_);
        drawStacked(result);
        console.log(result);
    }
}

app();
```

FIGURE 3.17. This code is the total JavaScript code for prediction.



```
/*  Template Style
    Edit this section*/

body {
  font-family: Arial;
  line-height: 2;
  font-size: 16px;
  background: #fff;
  color: #848484;
  font-weight: 300;
  overflow-x: hidden;
}
body.fh5co-offcanvas {
  overflow: hidden;
}

a {
  color: #ff5722;
  -webkit-transition: 0.5s, ease;
  -o-transition: 0.5s, ease;
  transition: 0.5s, ease;
}
a:hover {
  text-decoration: none;
  color: #ff5722;
```

FIGURE 3.18. Basic structure of CSS

3.2.5 Css

Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents[2]. These pages contain information on how to learn and use CSS and on available software.

Chapter 4

Discussion

In this final chapter, we briefly explain several complications & limitations related to our project and how we can make the current system more efficient by doing further developments in the future. After that, the concluding parts of the report is included.

4.1 Complications and Limitations

For building an user interface based on the image processing model is now new. There are lots of work done and there are lots of example out there but during the implementation of this user interface we have faced lots of complication like, sometimes some files did found in the given model and there is few times it occurs that lots of file have undefined function and there is code error. To train the model in the user interface label we need the whole image data set. so when we were building our web application we have faced that it is very difficult to train a model in local environment. In that case we had to rely on the pre-trained model and we got some model some were worked properly. Some pre-trained model is not well performed.

4.2 Future Work

In future we will try to build a user dashboard, so that user can easily save their record in their home page.

4.3 Conclusion

The computational models, which were implemented in this project, were chosen after extensive research, and the successful testing results confirm that the choices made by the researcher were reliable.

this project began as an implementation of the Deep Art algorithm in recently open-sourced TensorFlow, an alternative and curious application of deep neural networks. This meant learning many new concepts on this field, and understand how those concepts could be translated to the new framework.

There are still many areas for research in this domain, as this project has just been an introductory implementation to build upon, now that a basic implementation is fully working. First, when the synchronous mode is fully implemented in the public version of TensorFlow it should be straightforward to implement, as basic tests were already done, but the inner functions of TensorFlow were not working as expected. Synchronous replicas open a new window of test, with the addition of backup workers, that should alleviate some of the drawbacks of synchronisation. Second, it would be interesting to test some application that exploits model parallelisation instead of data parallelisation. Although it is said that it does not scale as well as the latter and the configuration is pretty hard because operations and layers need to be placed manually, it could be an interesting experiment. Finally, it could be interesting to evaluate the effect of network congestion to training times. As the network of the cluster is shared among all the jobs, it is unclear if slowdowns during training could be caused by a congested network, and to which extend it affects the process.

Bibliography

[1] *Analitic vidhya for machine learning model*. Retrieved from `https://www.analyticsvidhya.com/`, Accessed: 2021-5-19.

[2] *Css programming language*. Retrieved from `https://www.w3schools.com/css/`, Accessed: 20120-9-20.

[3] *Google lens image checker*. Retrieved from `https://www.lens.google.com/`, Accessed: 2021-5-19.

[4] *Tensorflow model*. Retrieved from `https://https://www.tensorflow.org/js/models`, Accessed: 2021-5-19.

[5] *Tensorflow.js model git hub*. Retrieved from `https://github.com/tensorflow/tfjs-models/tree/master/mobilenet`, Accessed: 2021-5-19.

[6] *Tensorflow.js transfer learning image classifier*. Retrieved from `https://www.w3schools.com/js/`, Accessed: 20120-9-20.

[7] R. K. J. M. A Chougule, A Pawar, *Chougule, a., pawar, a., kamble, r., mujawar, j., bhide, a. recognizing fresh and rotten fruits using deep learning techniques. data engineering and intelligent computing: Proceedings of icicc 2020, 205.*, Informing Science 11, (2020), pp. 60–79.