Mushroom Image Classifier

Juan Garcia

Date: 6/30/2022

Table of Contents

[A.1 Letter of Transmittal – Mushroom Patch Co. 4](#_Toc108789984)

[A.2 Project Recommendations 6](#_Toc108789985)

[A.2.1 Problem Summary 6](#_Toc108789986)

[A.2.2 Application Benefits 6](#_Toc108789987)

[A.2.3 Outline of the Data Product 6](#_Toc108789988)

[A.2.4 Data Used in the Data Product 7](#_Toc108789989)

[A.2.5 Objective and Hypothesis 7](#_Toc108789990)

[A.2.6 Outline of Methodology 7](#_Toc108789991)

[A.2.7 Funding Requirements 7](#_Toc108789992)

[A.2.8 Impact of the Solution on Stakeholders 8](#_Toc108789993)

[A.2.9 Ethical and Legal Considerations 8](#_Toc108789994)

[A.2.10 Developer’s Expertise 8](#_Toc108789995)

[B. Project Proposal for IT Professionals 8](#_Toc108789996)

[B.1 Problem Statement 8](#_Toc108789997)

[B.2 Customer Description and Benefits 9](#_Toc108789998)

[B.3 Existing Systems Integration 9](#_Toc108789999)

[B.4 Necessary Data 9](#_Toc108790000)

[B.5 Methodology 10](#_Toc108790001)

[B.6 Deliverables 11](#_Toc108790002)

[B.7 Implementation Plan 11](#_Toc108790003)

[B.8 Evaluation Plan 12](#_Toc108790004)

[B.9 Programming Environments and Related Costs 12](#_Toc108790005)

[B.10 Timeline and Milestones 12](#_Toc108790006)

[D. Developed Product Documentation 13](#_Toc108790007)

[D.1 Project Purpose 13](#_Toc108790008)

[D.2 Raw and Cleaned Data 13](#_Toc108790009)

[D.3 Code Analysis 15](#_Toc108790010)

[D.4 Hypothesis Verification 17](#_Toc108790011)

[D.5 Effective Visualizations and Reporting 18](#_Toc108790012)

[D.6 Accuracy Analysis 19](#_Toc108790013)

[D.7 Application Testing 19](#_Toc108790014)

[D.8 Application Files 19](#_Toc108790015)

[D.9 User’s Guide 20](#_Toc108790016)

[Access via Heroku (Streamlit Implementation): 20](#_Toc108790017)

[Access via Google Colab (Gradio Implementation, this method requires a Google Account): 23](#_Toc108790018)

[D.10 Summation of Learning Experience 27](#_Toc108790019)

[E. References 28](#_Toc108790020)

# A.1 Letter of Transmittal – Mushroom Patch Co.

Dear Director Edwin Morales

Mushroom Patch Co. is one of the leading retail distributors of mushroom grow supplies and equipment geared towards amateur and professional mycologists and enthusiasts. Mushroom Patch Co. is seeing unprecedented growth in the past year and is expected to continue that growth in the next year. With this, an increasing number of customers are visiting the MPC website. This influx of users has caused some congestion in the MPC forums section of the site. An overwhelming amount of MPC users have asked for assistance identifying mushroom species. While allocating additional human resources to resolve customer issues and requests may be the simplest solution, there are concerns with the long-term costs of hiring such staff. Also, due to the increasing number of users, staff would inevitably face the same issue. Therefore, there is a great need for an efficient and automatic solution.

The proposed solution is to create software that can be accessed through a web browser. The web application would be able to receive images from users and reveal what kind of mushroom is in the image. This is accomplished using machine learning. Machine learning enables our software to be trained to identify mushrooms in images with high accuracy. It also allows our software to efficiently scale in the long-term since our machine learning solution works extremely well with a large amount of data. This software will be deployed over the web and will be very easy to use and intuitive. The web app is focused on ease of use and therefore will have a simplistic user interface. The web app will give the user to upload an image, display the image back to the user, and identify the mushroom by name. The image is loaded into the machine learning solution, and it will predict what type of mushroom is in the image based on its training. The program will be called Mushroom Image Classifier. Images of flowers will train the machine learning solution. These images will be obtained from the free dataset website Kaggle.com. The specific dataset we have obtained will consists of 6 different species of mushrooms that are very commonly found in the United States. One of the benefits of our machine learning solution is that it can be modified to retrieve additional data via web scraping. This can be completed in the future with additional development time. In addition, the images the MPC users provide will also help improve the accuracy of the machine learning algorithm.

The Mushroom Image Classifier will prove to be an effective way to satisfy user requests that flood the MPC forums. It will also be a valuable tool to new and old MPC customers and can be a potential avenue for future advertisements with further development. This will reduce the need to hire additional resources dedicated to helping customers and will improve the productivity of employees by having them focus on other duties. The budget for this software will be minimal as this will be developed in house and utilize open-source libraries. Since the solution will be web-based, additional hardware or upfront hardware cost is not necessary. The project does not have any impact on the MPC physical location. In fact, due to the value the Mushroom Image Classifier provides, the tool may become popular and attract potential customers. There are little to no legal or ethical ramifications. The program does not collect any sensitive or personal user data in the long-term. The initial dataset is gathered from Kaggle.com, and the images gathered from guests are only stored temporarily to improve the algorithm’s training, and then immediately deleted.

My experience to manage this project consists of my education. I have an AS in Computer Science and have taken several courses towards my BS. My education required me to develop several programs for coursework and will help in developing a ML solution and an intuitive user interface for users.

Thank you for your time,

Juan Garcia

# A.2 Project Recommendations

## A.2.1 Problem Summary

Mushroom Patch Co is a retail distributor of mushroom grow supplies and equipment. The number of mycologists and mushroom enthusiasts that visit the MPC forums have steadily increased and has caused forum-wide congestion. While MPC has staff answer customer requests through the forums, a long-term solution is needed to satisfy customer requests and decongest the MPC forums. Developing a web application that utilizes machine learning will help solve this issue and provide a valuable tool that MPC can offer to its current and potential customers.

## A.2.2 Application Benefits

First and foremost, MPC staff cannot reasonably respond to all forum requests, nor can it dedicate sufficient human resources for such tasks. The Mushroom Image Classifier will resolve users’ inquiries and reduce the strain on existing human resources. Secondly, developing such a tool would greatly benefit MPC in the long-term. It will reduce costs incurred by site congestion, increase customer satisfaction, and encourage customer retention, and can potentially provide an avenue for MPC advertisement via the web interface since it can be made available to the public.

## A.2.3 Outline of the Data Product

The web application can be accessed by visiting a specific website. The program is equipped with a simplistic graphical user interface and prompts the user to upload an image of a mushroom. Once the user submits an image, the program will then utilize a machine learning model that was trained with thousands of mushroom images to predict the species of mushroom that is present in the user-submitted image. Once it is complete, the prediction is displayed to the user. More information can be found in the following sections.

## A.2.4 Data Used in the Data Product

The initial dataset that the machine learning utilizes for training and validation was gathered from Kaggle.com. The data is unstructured and is comprised of 6 different mushroom species that are commonly found in the United States. The 6 mushroom species consists of: Amanita bisporigera, Amanita muscaria, Boletus edulis, Cantharellus, Omphalotus olearius, and Russula mariae. The mushroom dataset is separated in folders by their distinct class. The dataset is split into two subsets, training, and validation. Approximately 60% of the images will be used for training, and 40% for validation. Unfortunately, the classes do not have an equal number of images so the quantity of images available for a specific class varies per species. The dataset consists of 606 images of Amanita Bisporigera, 367 images of Amanita Muscaria, 444 images of Boletus Edulis, 1183 images of Cantharellus, 59 images of Omphalotus Olearius, and 235 images of Russula Mariae. The dataset is split into subsets so that we can test the accuracy of the model based on the initial training set. Further training can be achieved by developing a web scraping tool. This is out of scope, but it is important to note.

## A.2.5 Objective and Hypothesis

The primary objective of this application is to provide a simple tool that accepts user-submitted images and provides an accurate classification based on the image. The project will reduce human processing time on incoming user inquiries. The model should have a minimum 70% accuracy when tested against known data.

## A.2.6 Outline of Methodology

The project will follow the waterfall project methodology since this project will be undertaken by a singular developer. There are clear goals in mind from the beginning of this project. Due to the large quantity of data available on Kaggle.com, testing can be done during and after the project meets completion.

## A.2.7 Funding Requirements

The proposed software solution utilizes publicly available, open-source tools and libraries. Moreover, since MIC is a web-based application, there are no upfront hardware costs. The end-users utilize their own hardware to access the web application and therefore do not require Mushroom Patch Co. to install any additional hardware to any physical locations. The project’s primary cost will be the cost of human resources dedicated to developing the software product. This project is estimated to require approximately 200 hours from project approval to deployment. This is estimated to cost approximately $8000 USD.

## A.2.8 Impact of the Solution on Stakeholders

The most noticeable impact the Mushroom Image Classifier will have is an increase in customer satisfaction. Staff productivity is also expected to increase since customer inquiries will be handled quickly and efficiently, which allows employees to focus on other issues that need to be addressed. Marketing campaigns can also advertise the tool to MPC customers and survey customers for additional features to be developed in the future.

## A.2.9 Ethical and Legal Considerations

The program utilizes public available data and open-source software in its development. The program is solely focused on mushrooms and can only identify mushroom species. The data submitted by users will not be used to extract external information, nor is the data stored in the long-term. At no point does the project employ the use of human subject, nor does it utilize datasets that contain personal, sensitive, or otherwise identifiable information. The MIC can only predict the name of the mushroom and does not classify mushrooms based on their lethality.

## A.2.10 Developer’s Expertise

The developer will have a degree in computer science and several years of experience in Python. They will have prior experience in developing open-source projects, project management, and creating simple, yet intuitive user interfaces.

# B. Project Proposal for IT Professionals

## B.1 Problem Statement

Recently, wild mushroom foraging has become an increasingly popular activity. Mycologists and mushroom enthusiasts need a tool to efficiently identify species of mushrooms that are commonly throughout the United States. There is a need for an easy-to-use tool that can provide a moderately accurate prediction in a short amount of time.

## B.2 Customer Description and Benefits

The application’s primary target users will consist of Mushroom Patch Co. customers and mushroom enthusiasts. There are no restrictions on who can utilize this tool so non-customers can experience the tool.

## B.3 Existing Systems Integration

There are currently no systems in place that helps alleviate user identification inquiries. Staffs provide their own predictions manually, and therefore the general accuracy of the staff cannot be reasonably measured. By automating this task, staff will experience greatly reduced pressure to handle user inquiries and can dedicate time to other important issues.

## B.4 Necessary Data

The dataset consists of numerous images (unstructured data). There are approximately 2800+ images in the dataset. We expect to use 40% for validation. This data set consists of 6 different species (6 different classes) with quantities specified.

* Amanita bisporigera - 606
* Amanita muscaria - 367
* Boletus edulis - 444
* Cantharellus – 1183
* Omphalotus olearius - 59
* Russula mariae - 235

The second dataset consists of 8781 images of mushrooms. 246 images were extracted from this dataset and merged into the first dataset. The extracted data consisted of the following:

* Amanita Bisporigera – 48
* Amanita Muscaria – 51
* Boletus Edulis – 64
* Cantharellus – 50
* Omphalotus Orealius - 33

The data we are utilizing is available on Kaggle. Available at the following url: <https://www.kaggle.com/datasets/harperd17/mushroom-pictures>

<https://www.kaggle.com/datasets/derekkunowilliams/mushrooms>

Chart, bar chart

Description automatically generated

## B.5 Methodology

This project will be built utilizing the waterfall methodology. The project’s main purpose is multi-class image classification. The project requirements are clear and concise and unlikely to change. Evaluation of the program will be determined by the accuracy of the output. The project phases are described below:

1. **Phase 1:** Project Requirements -Gather and understand all project requirements upfront. Budget for the project should be proposed and the project accepted. Any necessary hardware should be obtained in this phase.
2. **Phase 2:** Project Design – Create an outline for the software architecture and design. Research and selection of effective machine learning algorithms should be done. Data collection/gathering should also be performed.
3. **Phase 3:** Project Implementation – Start development of the machine learning solution and build the software application. Three different architectural neural network models will be created and trained for initial evaluation. Speed of predictions, throughput, and accuracy will be assessed.
4. **Phase 4:** Project Testing – A model will be chosen, and a portion of the data will be used to perform further testing to evaluate the accuracy of the program. The GUI will also be designed for ease of use, and implemented at a local level.
5. **Phase 5:** Project Deployment – Deploy the software application to a cloud-based platform and analyze performance.
6. **Phase 6:** Project Maintenance – Monitor performance and perform additional training as needed.

## B.6 Deliverables

The deliverables will include the source code for training the models and the front-end portion of the software application. The program will be deployed on Heroku and is accessible via any modern web browser. All files related or involved with the creation of the program can be provided if requested.

## B.7 Implementation Plan

There are 4 major sections to the creation of this program: Data gathering and preparation, model creation, front-end development, and web deployment. While they can be worked on independently, all are necessary for the success of this project. We will use Anaconda Navigator to easily create a local runtime environment to work in Jupyter Notebook, Spyder as the IDE. I will also use Google Colab for additional resources. This way a model can be trained on my personal computer and another model can be trained using Colab resources.

## B.8 Evaluation Plan

Each section of the application will be thoroughly tested before being introduced to previous sections and reevaluated once integrated. This is to ensure that all pieces are working as intended. To consider our software a success, the program must allow the user to submit an image of a mushroom, predict the species successfully, and display the prediction to the user along with any relevant metrics. The datasets do not contain a csv with file names and classes, so a file will need to be created.

## B.9 Programming Environments and Related Costs

The programming environment will mostly focus on Anaconda and Spyder. Our virtual environment will be created with Anaconda, Jupyter Notebook will be utilized to create the model, Spyder will function as the IDE for developing the front-end portion of the program using the Python programming language. Libraries in our virtual environment are listed in the “Requirements.txt” file provided in the Capstone Project folder. All tools and libraries are free to use and open source, there will be no upfront or recurring cost for the environment.

The notebook will be run on a local machine and on Google Colab so that multiple models can be built simultaneously. The notebooks will be mostly identical, only the file paths included in the code will be altered.

The development team consists of 1 software developer. Approximately 200 hours is expected for this project which will cost about $8,000 USD. Maintenance costs comprise of any upgrades, issue resolutions, and standard system maintenance at a rate of $30 per hour.

## B.10 Timeline and Milestones

06/01/2022 – The proposal is accepted.

06/15/2022 – A technical proof of concept is presented.

06/18/2022 – Testing begins.

06/30/2022 – Delivered

**Sprint Schedule**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sprint** | **Start** | **End** | **Tasks** |
| 1 | Date  06/01/2022 | Date  06/03/2022 | Project Planning and Setup |
| 2 | Date  06/01/2022 | Date  06/04/2022 | Data Gathering and Preparation |
| 3 | Date  06/05/2022 | Date  06/15/2022 | Modeling |
| 4 | Date  06/18/2022 | Date  06/24/2022 | Testing |
| 5 | 06/25/2022 | 06/28/2022 | Evaluation |
| 6 | 06/28/2022 | 06/30/2022 | Deployment |
| 7 | 06/30/2022 | - | Monitoring and Periodic Retraining |

# D. Developed Product Documentation

## D.1 Project Purpose

The project was intended to provide a tool that can automatically predict a species of mushroom when a user submits an image. This was due to an overwhelming influx of users congesting the MPC forums and straining the available human resources. The ultimate goal of this project is to alleviate staff from such duties and provide a timelier response to customer inquiries in an effort to minimize customer dissatisfaction.

## D.2 Raw and Cleaned Data

The datasets were obtained from Kaggle.com. The datasets contained images of mushroom. There were very few images of Omphalotus Orealius so that class was omitted in the final model. Moreover, there was 1 image that was unable to be preprocessed in Jupyter Notebook and was deleted from the first dataset. The code for detecting any issues with the image files is provided at the bottom of the Jupyter Notebook file provided in the Capstone folder.

The datasets did not come with a csv file, so I manually created one using python command:

DIR | Export-Csv MyFileList.csv

The CSV file was necessary to associate filenames with class names.

Approximately 60% of images were used for training, 40% for validation due to the size of the dataset. Both datasets contained Omphalotus Olearius images, however this species was omitted since there was less than 100 images in the class folder. A data scraper could have been utilized to gather additional images, however that would require additional time that has not already been agreed upon.

For our input to work with Tensorflow, I had to turn the images into tensors. Text, letter

Description automatically generatedText

Description automatically generated

Once this was done, the data had to be turned into batches. We created our labels from the csv file we organized. There are 3 different types of batches we created, test, validation, and training data. The test data is assumed to not have any labels. If is a validation batch, then there is no need to shuffle any data. For the training batches, we want to shuffle the data to reduce variance, since we want our data sets to be representative of the overall distribution of data. This also makes sure our models overfits less.

Text

Description automatically generated

## D.3 Code Analysis

The code utilizes several libraries to help create the program and model. Necessary Conda Installs are listed in the Requirements.txt file included in the Capstone Project Folder. Tensorflow and Tensorflow hub were used to create the back end of the machine learning foundation. Keras was used as a wrapper for Tensorflow and to build the model file. Matplotlib was utilized to graph training and validation loss and accuracy. We constructed the application side of this project using Streamlit and deployed our program through the cloud-based platform Heroku. Further code analysis is provided in the source code Jupyter Notebook file provided in the Capstone Project Folder. Essentially, the model was developed using Tensorflow and Keras. Externally, the application was created with Streamlit, and loaded the model for proper use. Then once the application was built, it was deployed using Heroku. The model code can be found in the Jupyter Notebook file called “ImageClassifier.ipynb”, and the application code can be found in the python file called “predict\_image\_class.py”. A comparison of each model’s accuracy can be found in a later section. Several evaluations were performed throughout the creation of our final model. I created and trained a model on 2000 images before deciding to continue to train on the full set of ~3037. Functions were created to quickly determine the predictions of each model. Plot prediction confidence was to evaluate the prediction probabilities of the model trained on 2000 images.

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text, application, chat or text message

Description automatically generated Chart, waterfall chart

Description automatically generated

This was then used to quickly test a set of 9 random images in the validation set. A similar function was created for the model trained on the full data set.

A picture containing graphical user interface

Description automatically generated

Chart, bar chart

Description automatically generated

## D.4 Hypothesis Verification

The hypothesis for this program is as follows: Can machine learning be effective at predicting, with high accuracy, the species of a mushroom based on an image? According to the models we have created, the answer is yes. Using prebuilt models and further training them, we have been able to achieve +85% accuracy based on our MobileNetv3s model. In manual testing, we were able to correctly predict 33 out of 34 images. Therefore, I believe we have proved that that machine learning is a viable solution for our problem.

Text

Description automatically generated



## D.5 Effective Visualizations and Reporting

Chart, line chart

Description automatically generatedText

Description automatically generatedChart, line chart

Description automatically generatedText

Description automatically generated

Chart, line chart

Description automatically generatedText

Description automatically generated

Matplotlib and Tensorboard allow for the monitoring of the model’s training during its 2 phases of creation. The model was initially trained on a smaller quantity of images, approximately 2000, while the subsequent model was trained on the full set of data. It also allowed me to easily compare the 4 separate pretrained neural networks against each other during the separate phases. It also demonstrates at which point each model begins to overfit, since the models have an early stopping function to prevent overfitting.

## D.6 Accuracy Analysis

To consider the project a success, an accuracy above 80% is desired. During the training phase, the MobileNet model achieved 99% accuracy, 85% during the validation phase. When I tested the model on a set of 34 images, the model correctly guessed 33 out of the 34 images, 97%. The program has achieved acceptable accuracy. Accuracy and loss graphs can be found in earlier sections.

## D.7 Application Testing

The application was also tested on various devices to ensure functionality. Since the notebook and the web application are separate files, each new model had to be loaded into the python application to test if the application experienced any errors when calling for a prediction.

Several sections of the jupyter notebook file were tested before being used in following code segments. This is demonstrated throughout the notebook provided in the Capstone folder.

Acceptance Testing was performed to ensure the application was easy to use, functional, and accurate. This was done by sharing a link to the Heroku application to a non-developer.

## D.8 Application Files

The source code can be found in the Capstone Project Program folder. The notebooks are also available in Google Colab and can be ran.

The notebook that created and trained the model can be found at the following URL: <https://colab.research.google.com/drive/1DaV72Wekxg1Sp4m1svD1N5MKBGwVA2a6?usp=sharing>

The notebook that contains the Gradio implementation of the mushroom image classifier can be found at the following URL: <https://colab.research.google.com/drive/1kxXufKbPevp8Ewb1095ZJqBx5aDPLfeV?usp=sharing>

The Heroku application can be found at: <https://mush-image-classifier-2.herokuapp.com/>

The Heroku Streamlit Application repository can be found at the following URL: <https://github.com/Jugarcia15/Streamlit-Mushroom-Image-Classifier.git>

The Google Colab Retrieval Repo that is used in the model creation and training can be found at <https://github.com/Jugarcia15/Google-Colab-Retrieval-Repo.git>

The Google Colab Retrieval Repository contains the trained model, the reorganized image dataset, logs for tensorboard, and a folder of 34 test images.

The Capstone Submission Folder will include the following:

* Python Files:

1. Predict\_image\_class.py – Streamlit Front-end

* Ipython Files:

1. ImageClassifier.ipynb – Jupyter Notebook

* Additional Files:

1. MN\_Full\_model.h5 – MobileNet Model
2. Procfile, setup.sh – Heroku required files
3. Requirements.txt – required library installations
4. Sample\_image\_rm.jpg – test image

## D.9 User’s Guide

To use the application, users will only need access to the internet and any modern internet browser.

### Access via Heroku (Streamlit Implementation):

1. Use any modern web browser, go to <https://mush-image-classifier-2.herokuapp.com/>

Graphical user interface, application, Word

Description automatically generated

2. Click ‘Browse Files’ in the side bar.

Graphical user interface, text, application, chat or text message

Description automatically generated

3. Select a mushroom image you want to classify. Then click ‘Open’. There is a test image labeled ‘sample\_image\_rm.jpg’ in the Capstone Submission folder. It is an image of a Russula Mariae mushroom.

Graphical user interface

Description automatically generatedGraphical user interface, text

Description automatically generated with medium confidence

3. Click ‘Predict Image’ in the sidebar to predict the image you have uploaded.

Graphical user interface, text, application

Description automatically generated

4. Prediction results will be displayed underneath the site title.

A picture containing text, fungus

Description automatically generated

### Access via Google Colab (Gradio Implementation, this method requires a Google Account):

1. Use any modern web browser, go to <https://colab.research.google.com/drive/1kxXufKbPevp8Ewb1095ZJqBx5aDPLfeV?usp=sharing>
2. Sign into a Google Account. After signing in, your browser should look similar to below:

Graphical user interface, text, application, email

Description automatically generated

1. On the top right-hand side of the page, click ‘Connect’ to connect to Colab runtime. You will be connected once you see ‘RAM’ and ‘Disk’ replace the ‘Connect’ button.

Graphical user interface, text, application, chat or text message

Description automatically generated Graphical user interface, application

Description automatically generated

1. Once Connected, click on ‘Runtime’ on the top left, and select ‘Run All’ from the drop-down menu.

Graphical user interface, text, application

Description automatically generated Graphical user interface, text, application, email

Description automatically generated

1. You will receive this warning from Google. Click ‘Run Anyway’ to proceed. The Colab notebook will clone files from a personal GitHub repo that contains the trained model and a folder of test images in zipped format. The notebook immediately unzips them. The files are temporarily stored in the virtual runtime environment and will be deleted once the user disconnects. The files can be found in the sidebar on the left.

Graphical user interface, text, application, letter, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

1. Scroll down to the last cell, the cell containing the Gradio interface should be running. The notebook should look like this:

Graphical user interface, text

Description automatically generated

1. Drag and drop or click ‘Click to Upload’ an Image in the image box in the Gradio Interface.

Graphical user interface, application

Description automatically generated

1. Select an image to upload and click ‘Open’.

Graphical user interface

Description automatically generated

1. Click ‘Submit’ to submit your image and generate predictions based on that image.

A picture containing text, fungus

Description automatically generated

1. Image predictions will be displayed on the right in the ‘output’ section.

Graphical user interface, text, application

Description automatically generated

1. Click ‘Clear’ to start over from step 7 or click the cell stop button located on the top left area of the cell.

Graphical user interface, text, application

Description automatically generated

1. Click the down arrow located next to where the ‘Connect’ button was located and select ‘Disconnect and delete runtime’ in order to conclude the session.

Graphical user interface, text, application

Description automatically generated

# D.10 Summation of Learning Experience

At the onset of this project, I knew that I had to a lot of research to do since I have never created a program that utilizes machine learning in any sort of capacity. I learned how to create virtual environments through Conda, utilize various open-source libraries that I have never used before, and deploy a program through a cloud platform. I followed many tutorials through the course of this project, I learned how to switch from working in Google Colab to Jupyter Notebook and how to properly access files in both. I ran into several issues due to incompatibility between deprecated library installations, then figured out how to solve them.

I learned how to use pre-trained models from TensorFlow Hub, how to augment data if needed. I also learned how to prevent overfitting by early stopping a model in training. I learned the importance of adjusting the sizes of validation and test sets.

I had a lot of problems during the project. I had to rewrite many sections of the code, rebuild the models due to small errors that caused major issues. This project showed me how important it is to make sure you dedicate time to properly research and plan out the project, and that it is sometimes necessary to scrap and restart. Having clear and concise documentation is imperative to the success of the project. While I did not have much experience coding prior to pursuing a degree in Computer Science, this project has shown me that the many courses I have taken thus far has been a valuable and worthwhile experience. I’m excited to see what I’m able to produce in the near future, and what I’ll be able to learn from it as well.

# E. References

No outside sources were used.