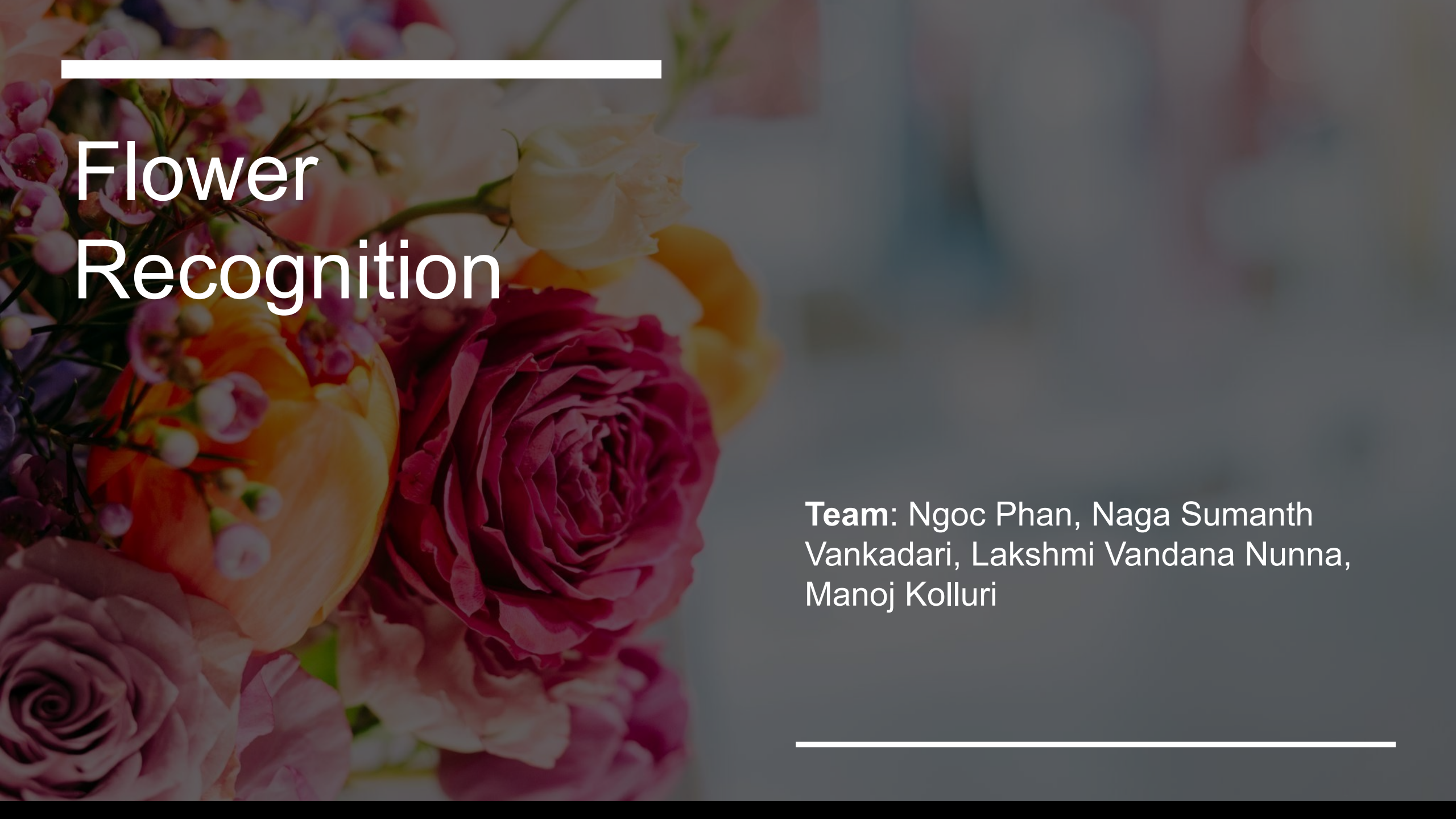
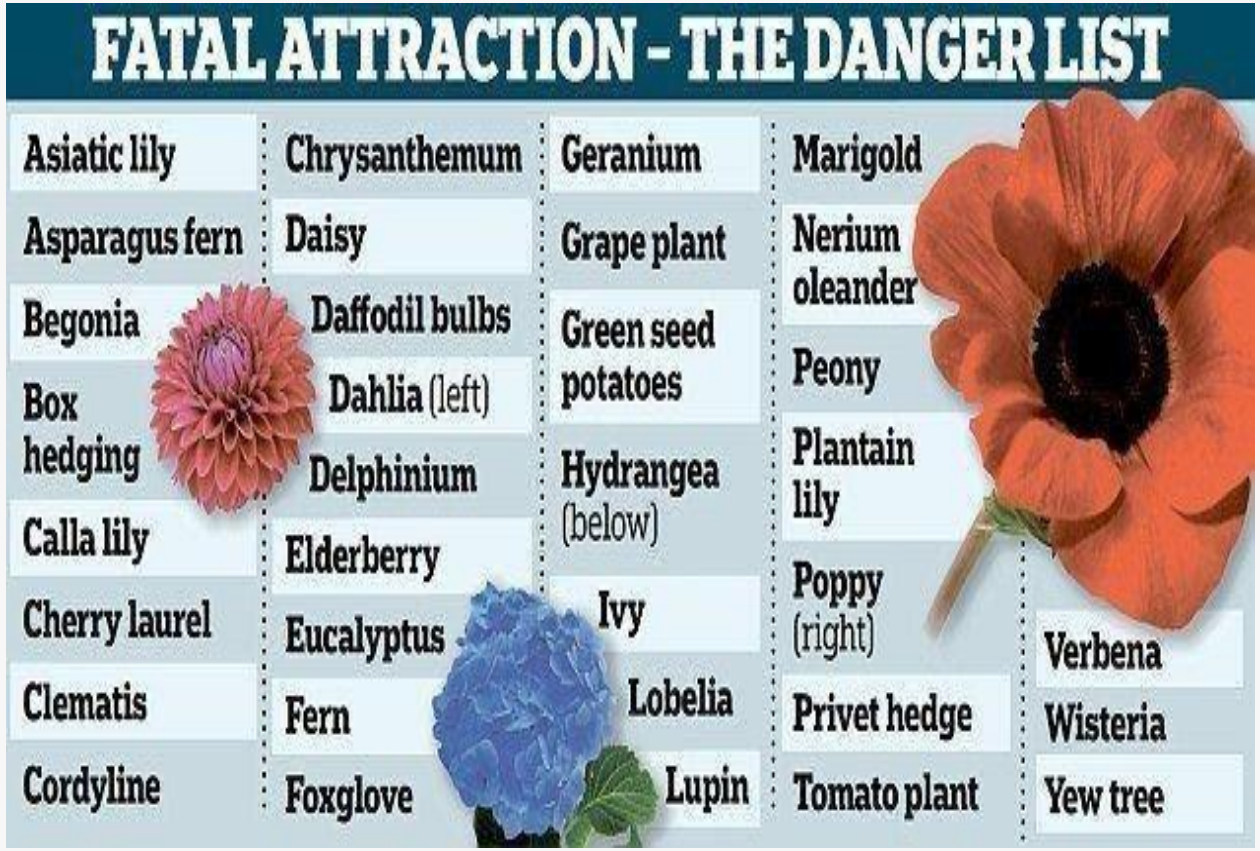


Flower Recognition

Team: Ngoc Phan, Naga Sumanth Vankadari, Lakshmi Vandana Nunna, Manoj Kolluri

Introduction



Recognize/identify the type of the flower given in the input image.

User-friendly front end integrated with 5 fully trained models

Resultant prediction labels from the models are displayed on the front end.

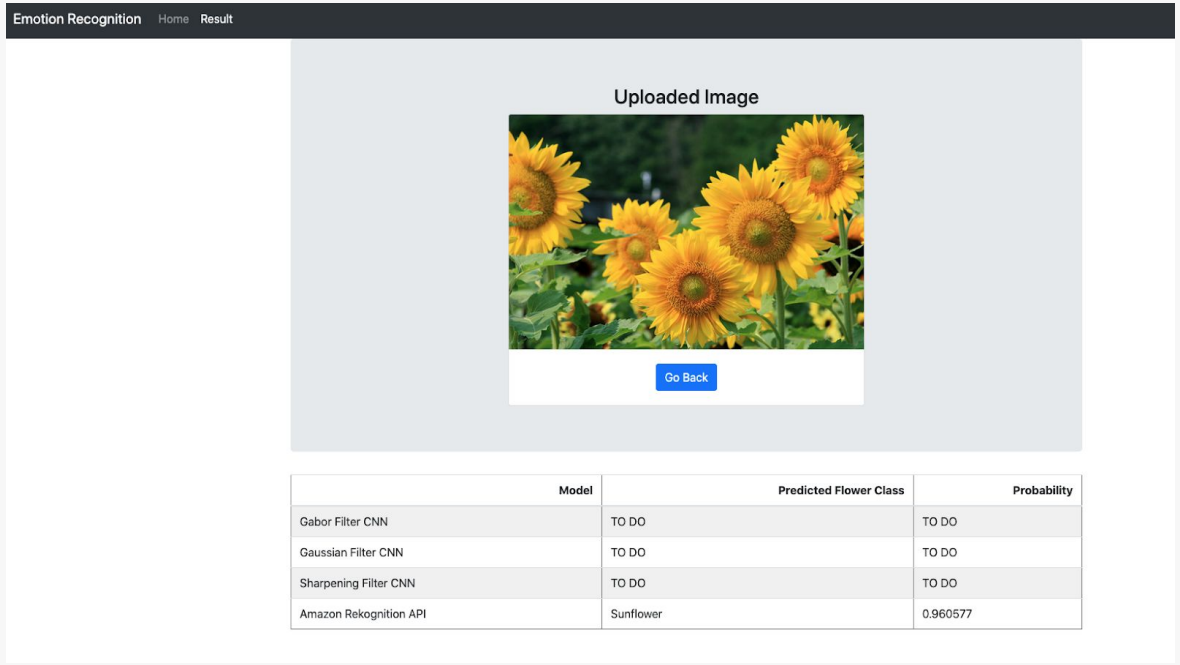
Motivation

Certain flowering plants release enzymes that trigger allergic reaction in humans.

They also attract poisonous bugs that could attack the human in the vicinity of the plant.

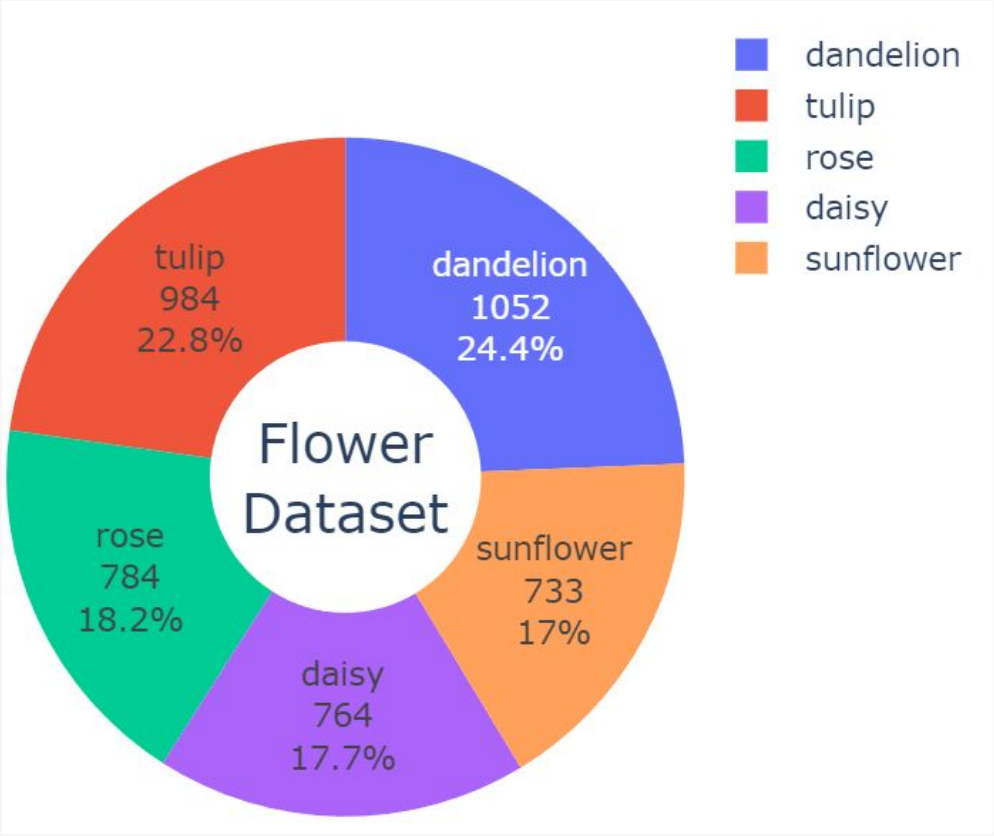
Helps in identifying and relating new plants to other identified plant species- this could contribute to research efforts in the fields of pollination, cloning, medicine, evolution study, plant science and lead to breakthroughs.

Goals

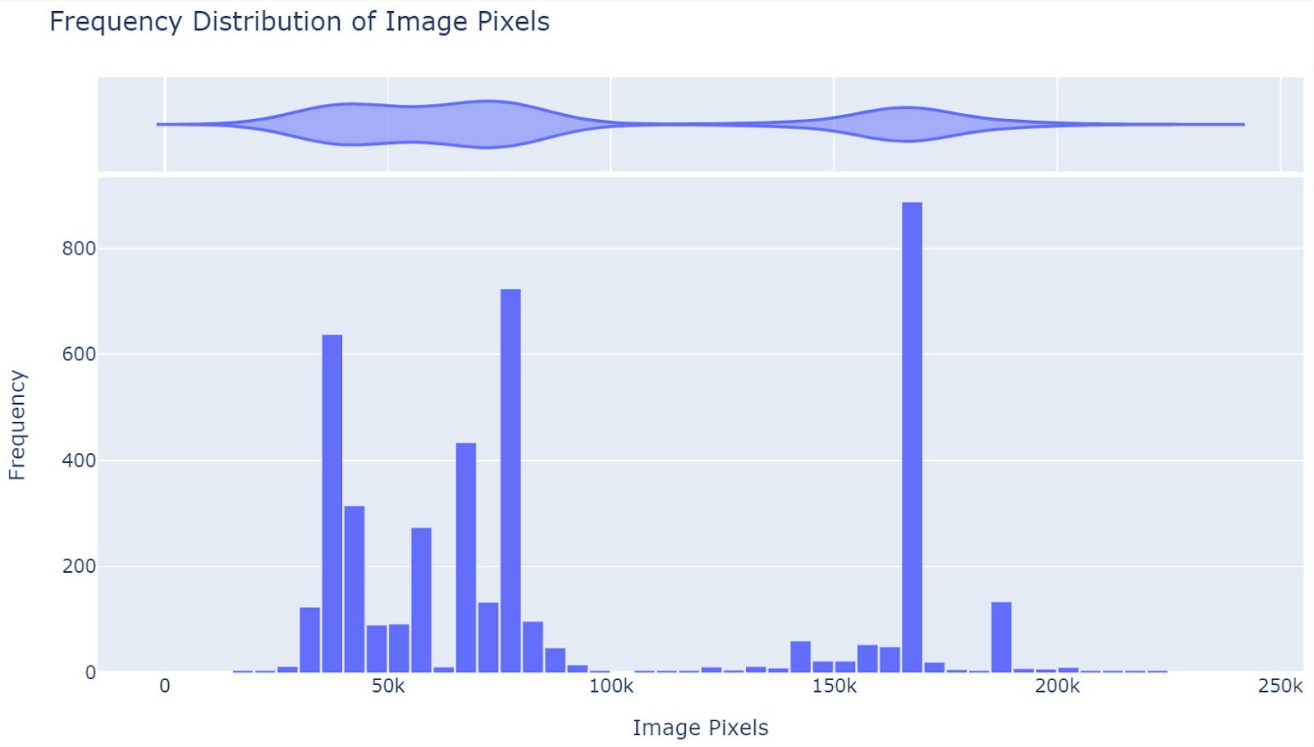


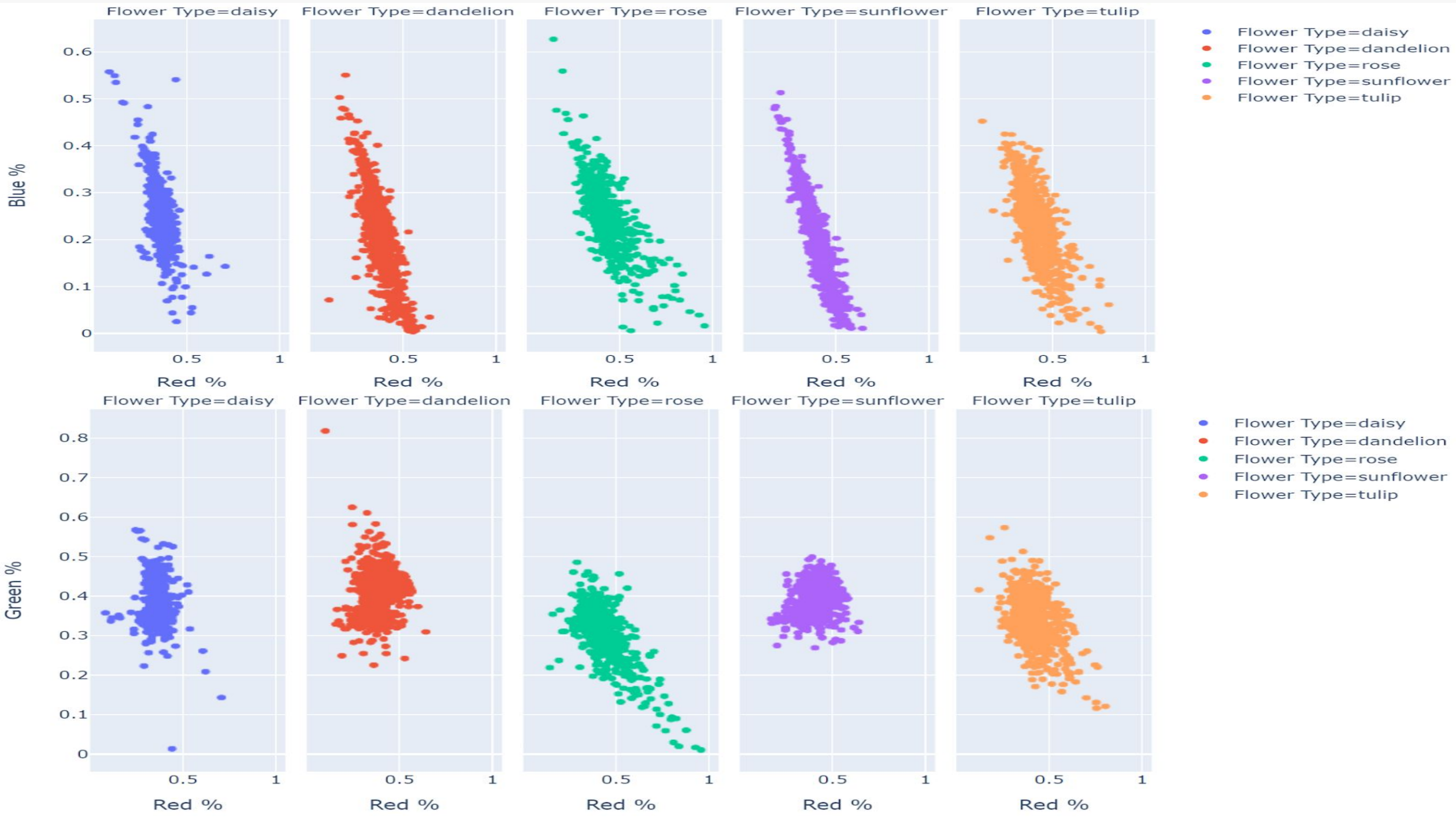
- GUI to enable the user to upload an image.
- Fully trained model that takes in vectorized images and predicts the label.
- GUI to display the predictions and metrics of AWS Rekognition API and the trained models
- Three different Convolutional neural networks trained on image features like Gaussian, Sharp and Gabor respectively.
- A concatenated model that is composed of all the above 3 CNN models.

Flower Data

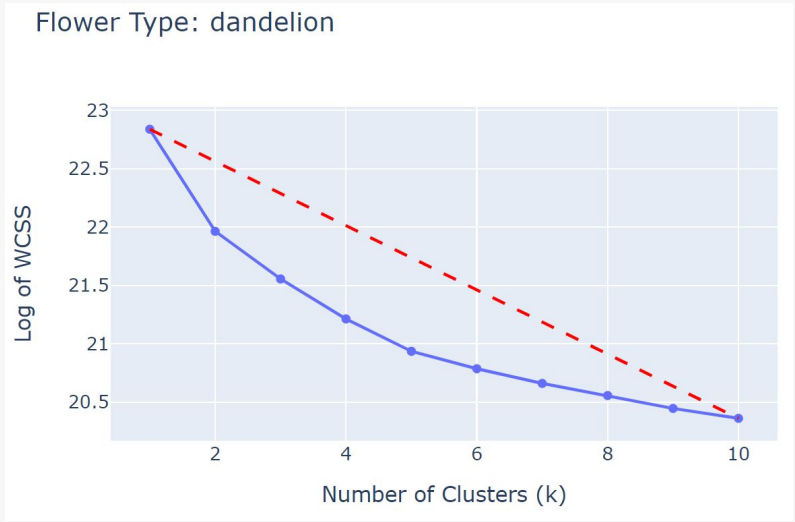


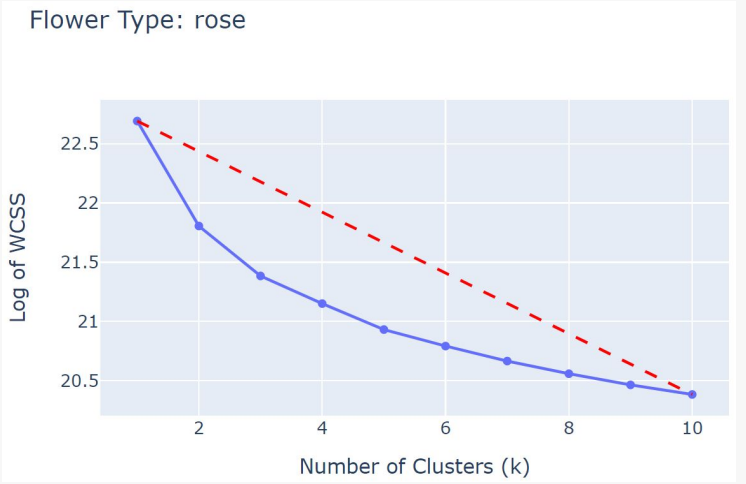
	Range	Min	Max	Mean	Median	Standard Deviation
Image Width	890.000000	134.000000	1024.000000	338.379893	320.000000	119.067232
Image Height	362.000000	80.000000	442.000000	253.073662	240.000000	61.558205
Image Pixels	201800.000000	19200.000000	221000.000000	91445.254112	76480.000000	52130.492116
Red %	0.888605	0.084733	0.973338	0.395124	0.377033	0.079271
Green %	0.807317	0.010713	0.818030	0.358717	0.355041	0.060148
Blue %	0.623759	0.003307	0.627066	0.246159	0.260725	0.083258

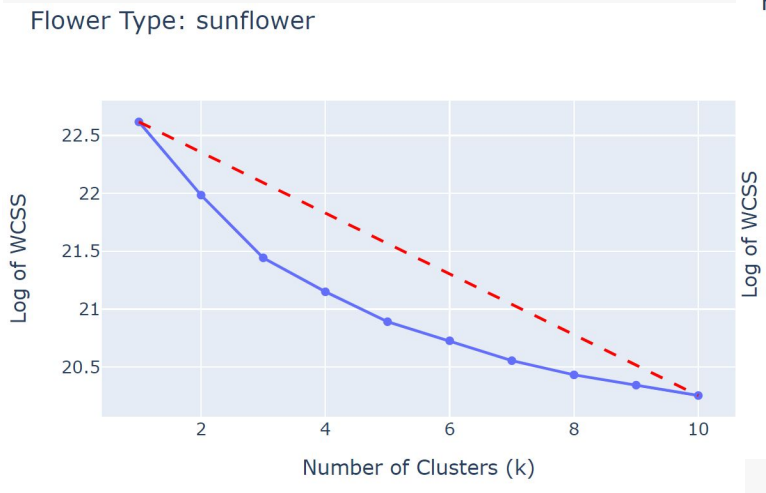


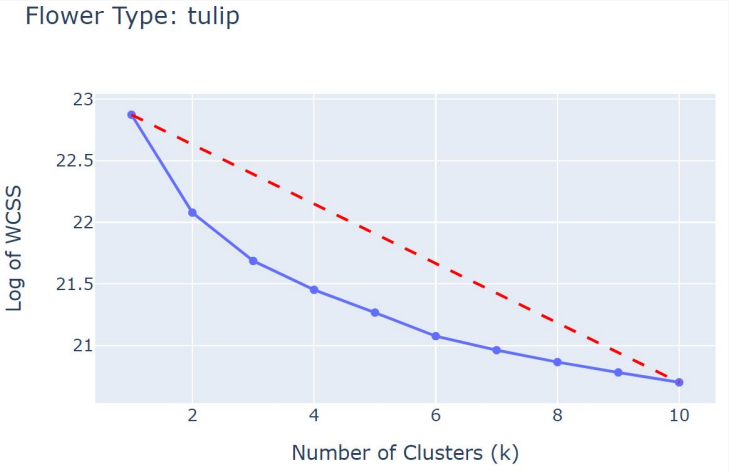

Flower Type=daisy
Flower Type=dandelion
Flower Type=rose
Flower Type=sunflower
Flower Type=tulip
Blue %
Green %
Red %
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
1

WCSS
vs
No of clusters









Dominant Colours for each category

dandelion

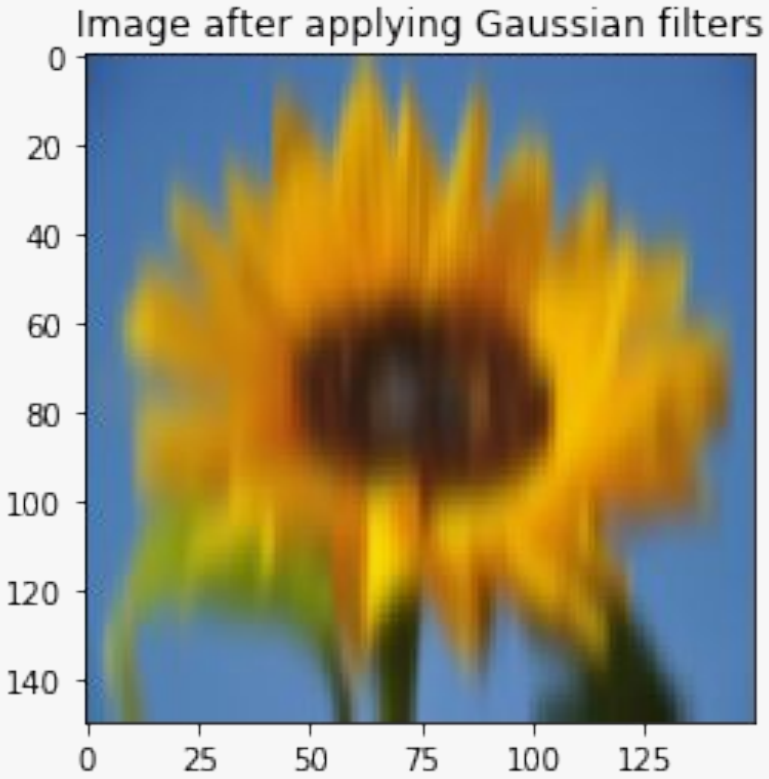
daisy

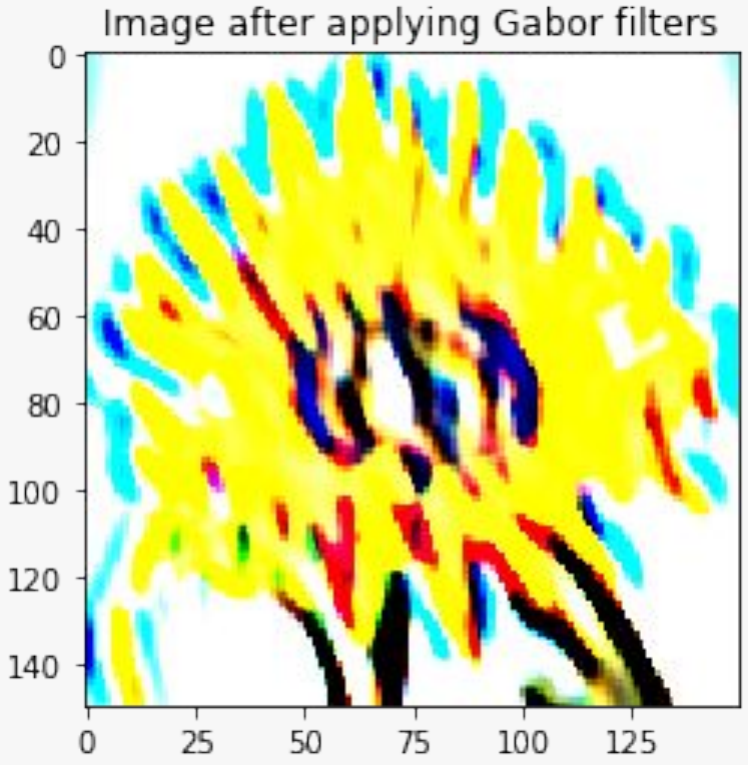
sunflower

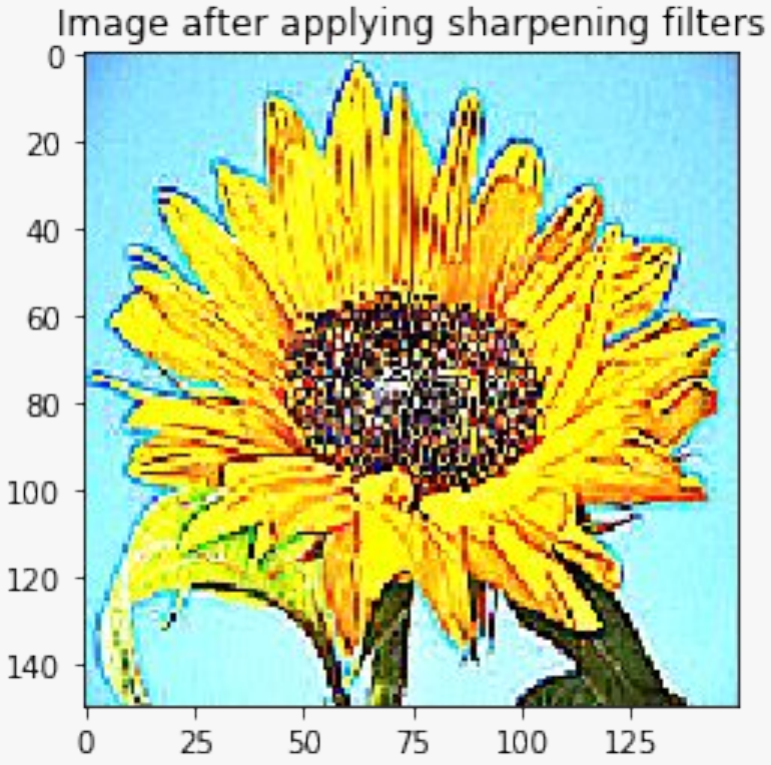
tulip

rose

Feature Extraction







CNN Models

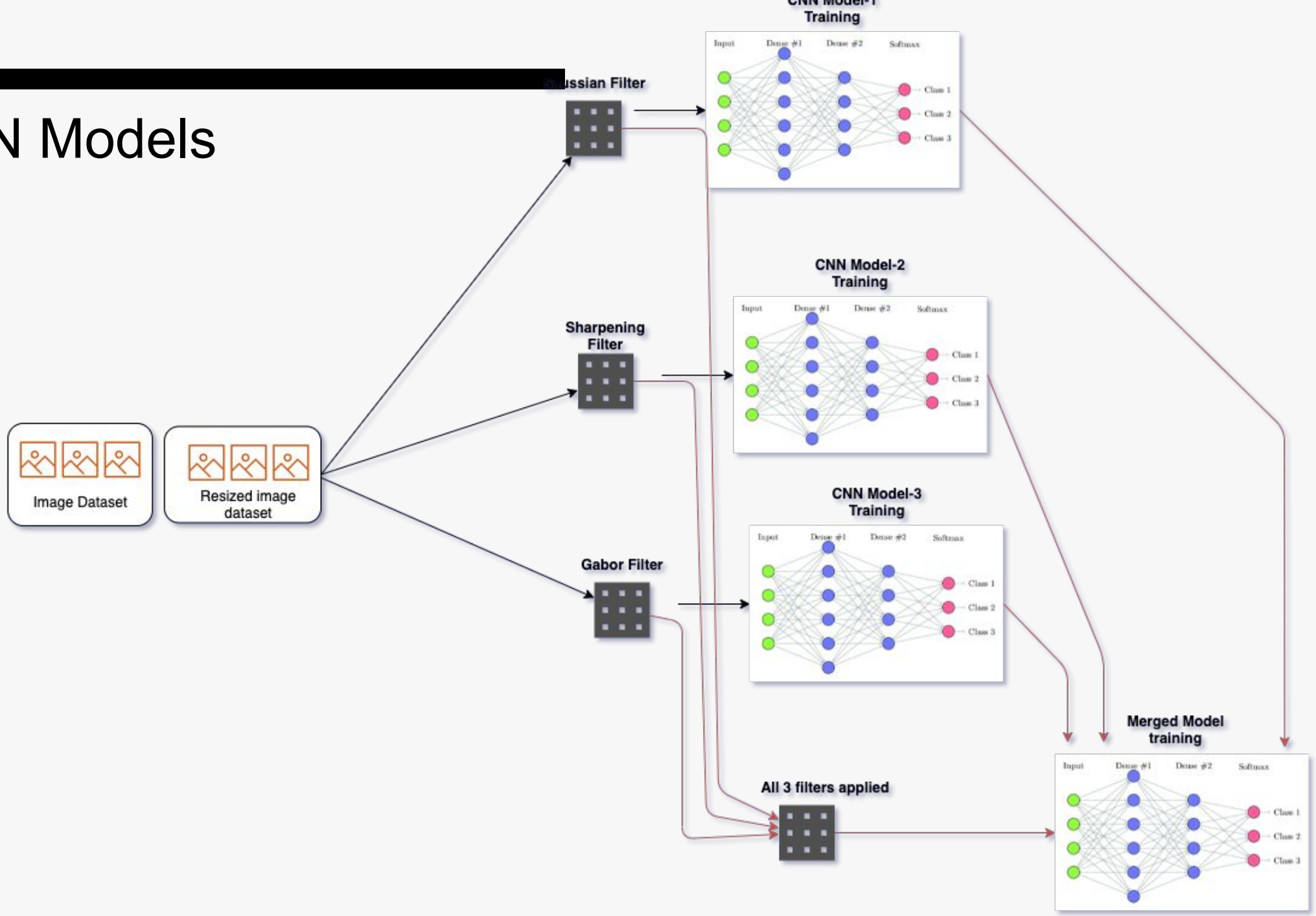
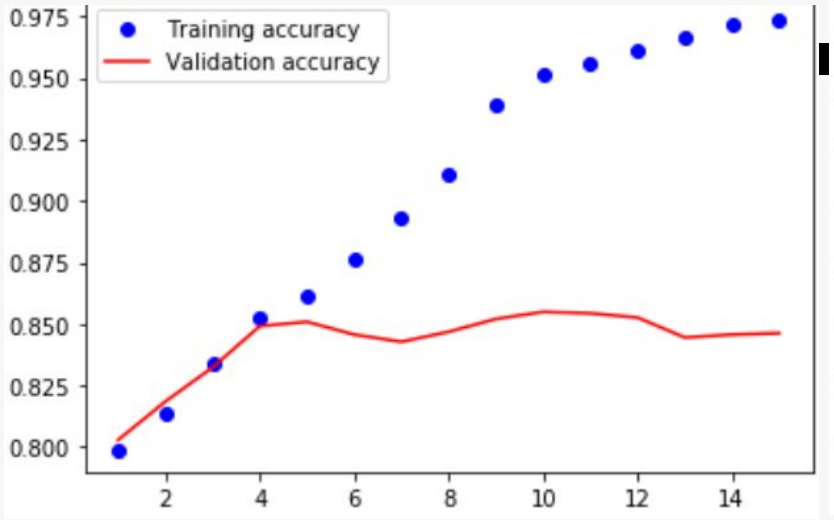


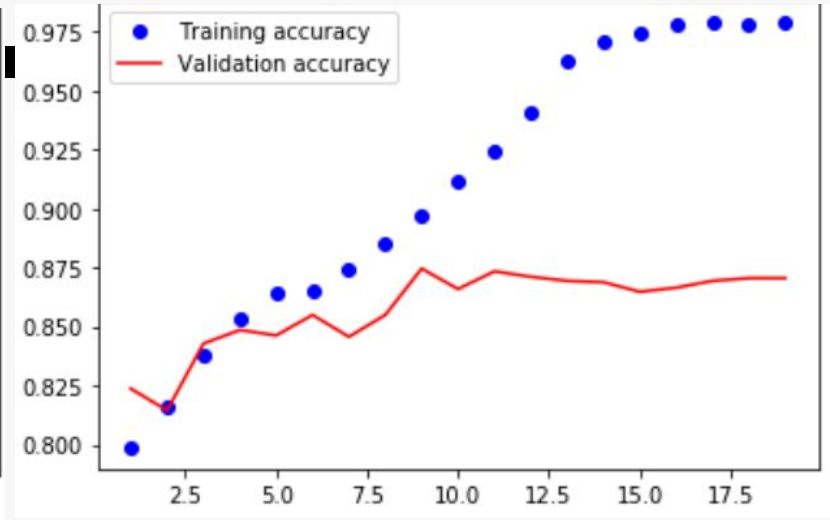
Image Dataset
Resized image dataset
Gaussian Filter
Sharpening Filter
Gabor Filter
CNN Model-1 Training
CNN Model-2 Training
CNN Model-3 Training
All 3 filters applied
Merged Model training
Input
Dense #1
Dense #2
Softmax
Class 1
Class 2
Class 3

CNN with Keras

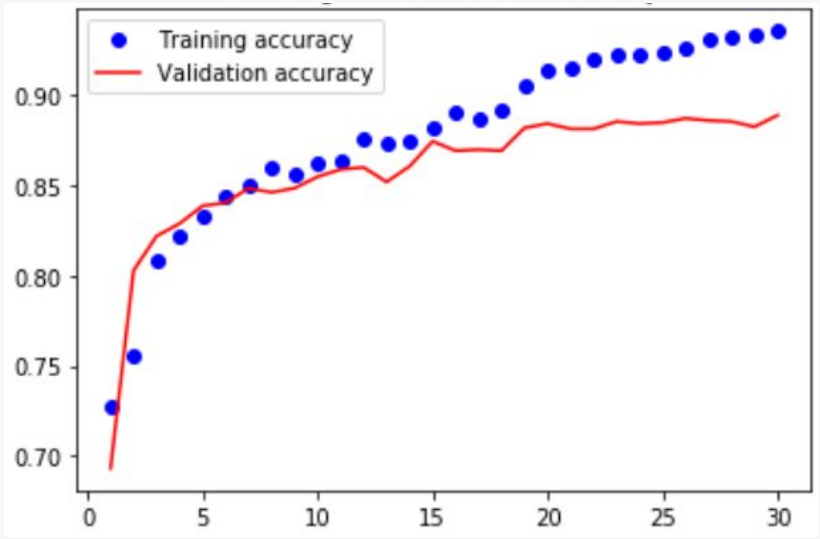
```
input1 = Input(shape=(size, size, 3))
x = Conv2D(filters = 32, kernel_size = 5, padding = 'same', activation = 'relu')(input1)
x = MaxPooling2D((2, 2))(x)
x = Conv2D(filters = 64, kernel_size = 3, padding = 'same', activation = 'relu')(x)
x = MaxPooling2D((2, 2),strides=(2,2))(x)
x = Conv2D(filters = 96, kernel_size = 3, padding = 'same', activation = 'relu')(x)
x = MaxPooling2D((2, 2),strides=(2,2))(x)
x = Conv2D(filters = 96, kernel_size = 3, padding = 'same', activation = 'relu')(x)
x = MaxPooling2D((2, 2),strides=(2,2))(x)
x = Flatten()(x)
x = Dense(512)(x)
x = Activation('relu')(x)
out = Dense(5, activation = "softmax")(x)

gabormodel = Model(input1, out)
gabormodel.compile(optimizer=Adam(lr=0.001),loss='binary_crossentropy',metrics=['accurac
y'])
```



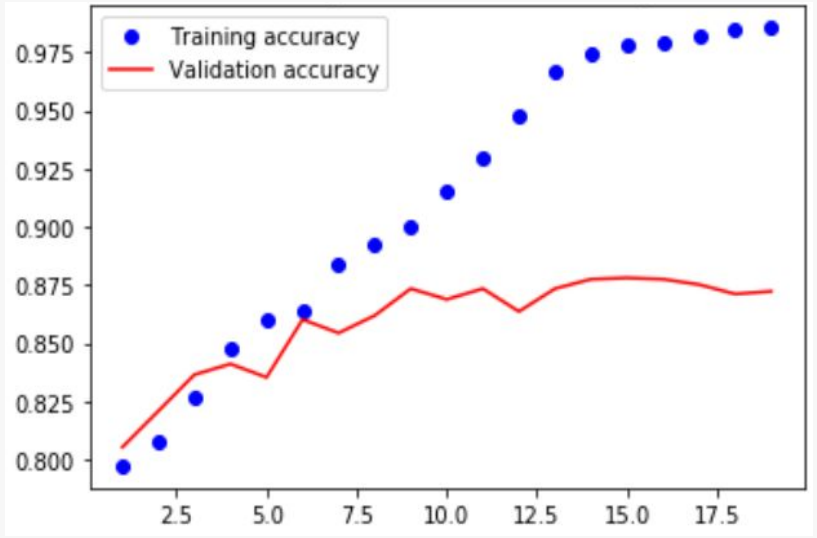
Gabor CNN



Gaussian CNN

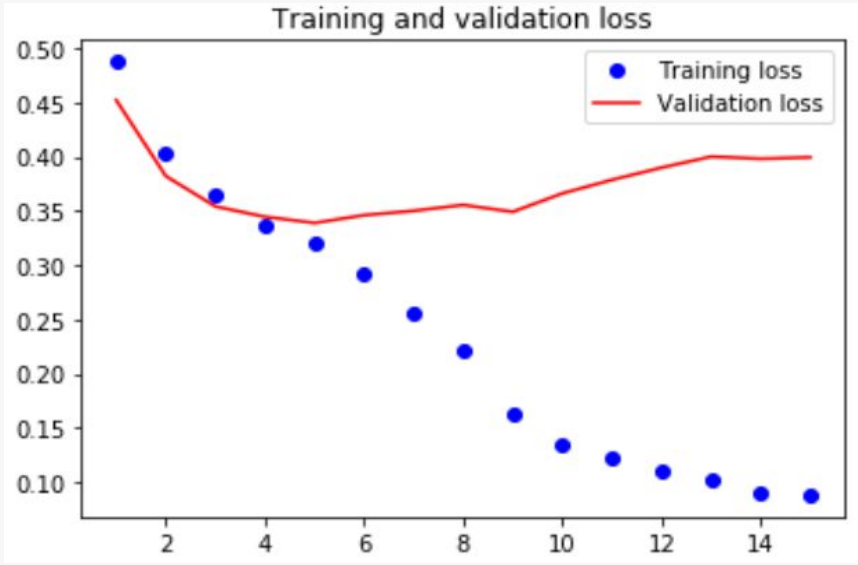


Merged CNN

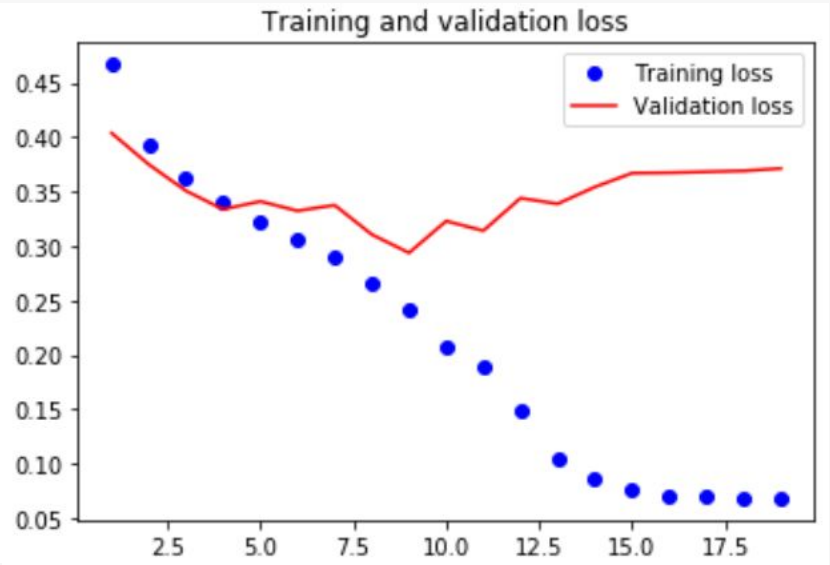


Sharpening CNN

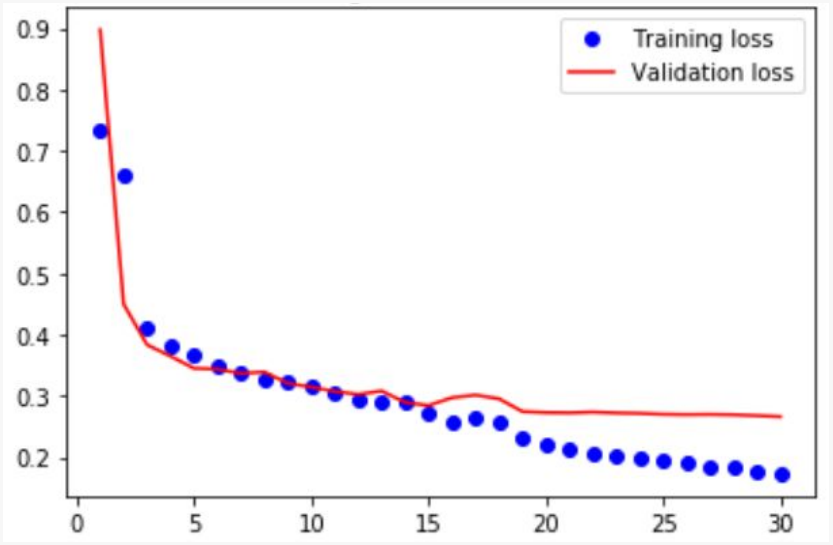
Training and validation accuracy



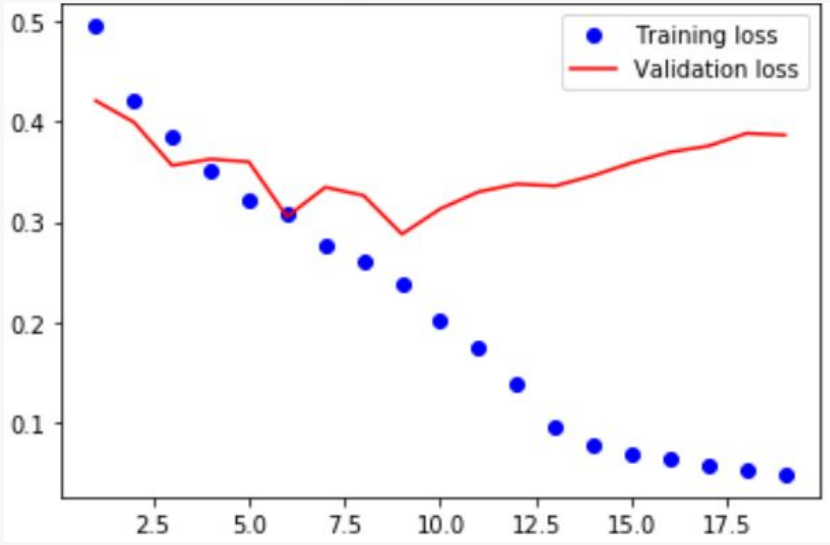
Gabor CNN



Gaussian CNN

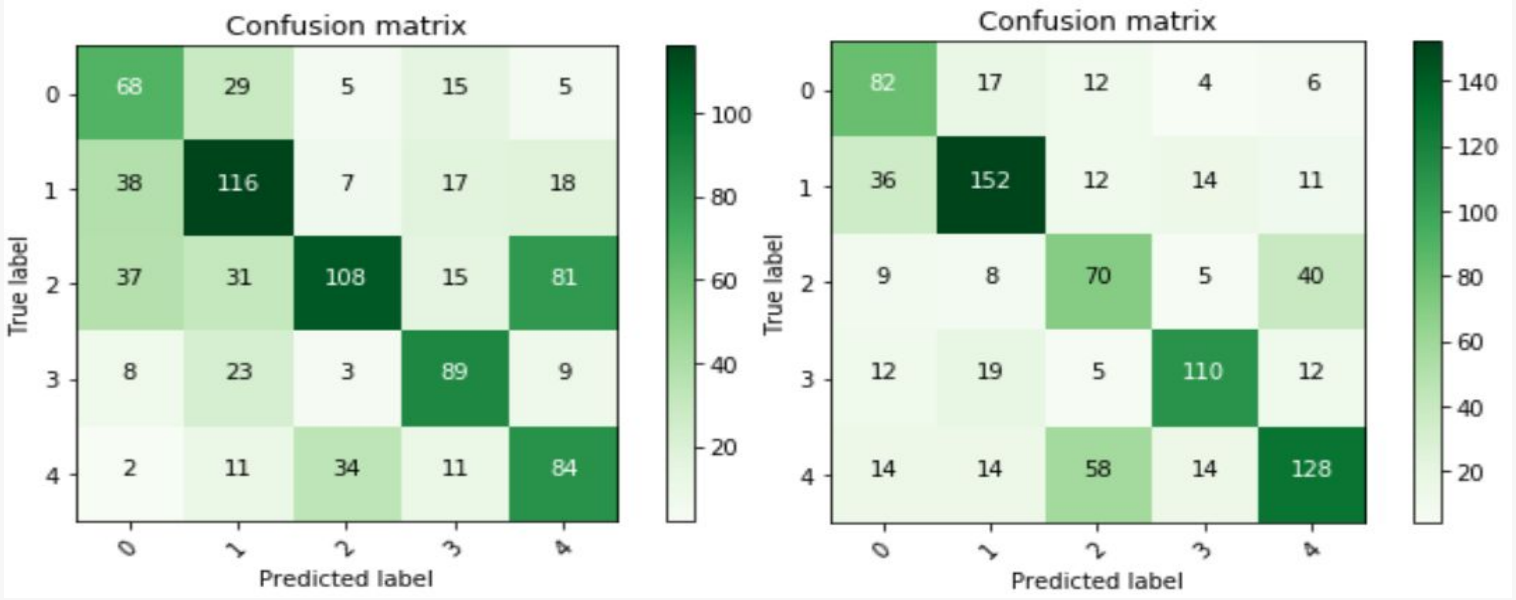


Merged CNN



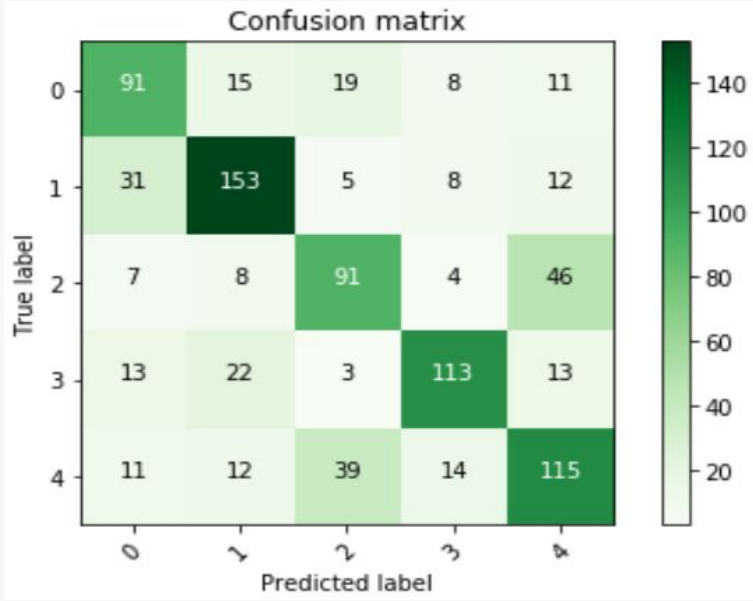
Sharpening CNN

Training and validation loss

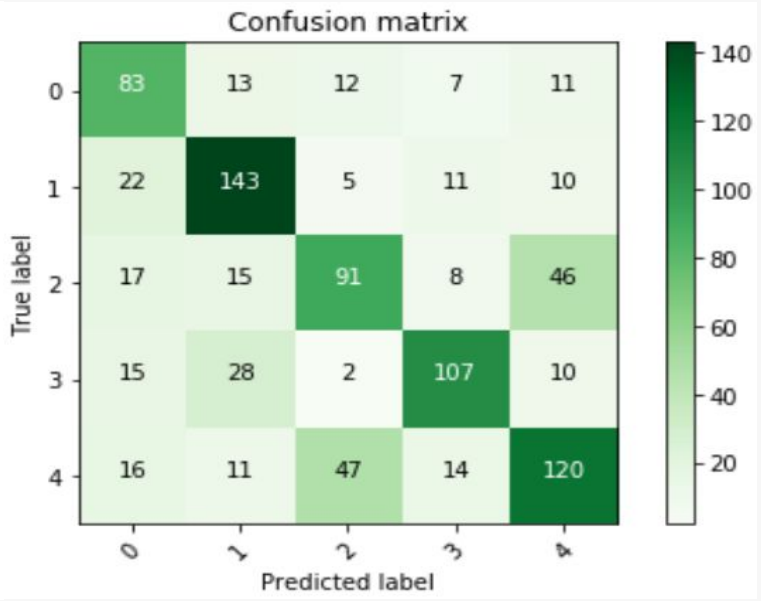


Gabor CNN

Gaussian CNN



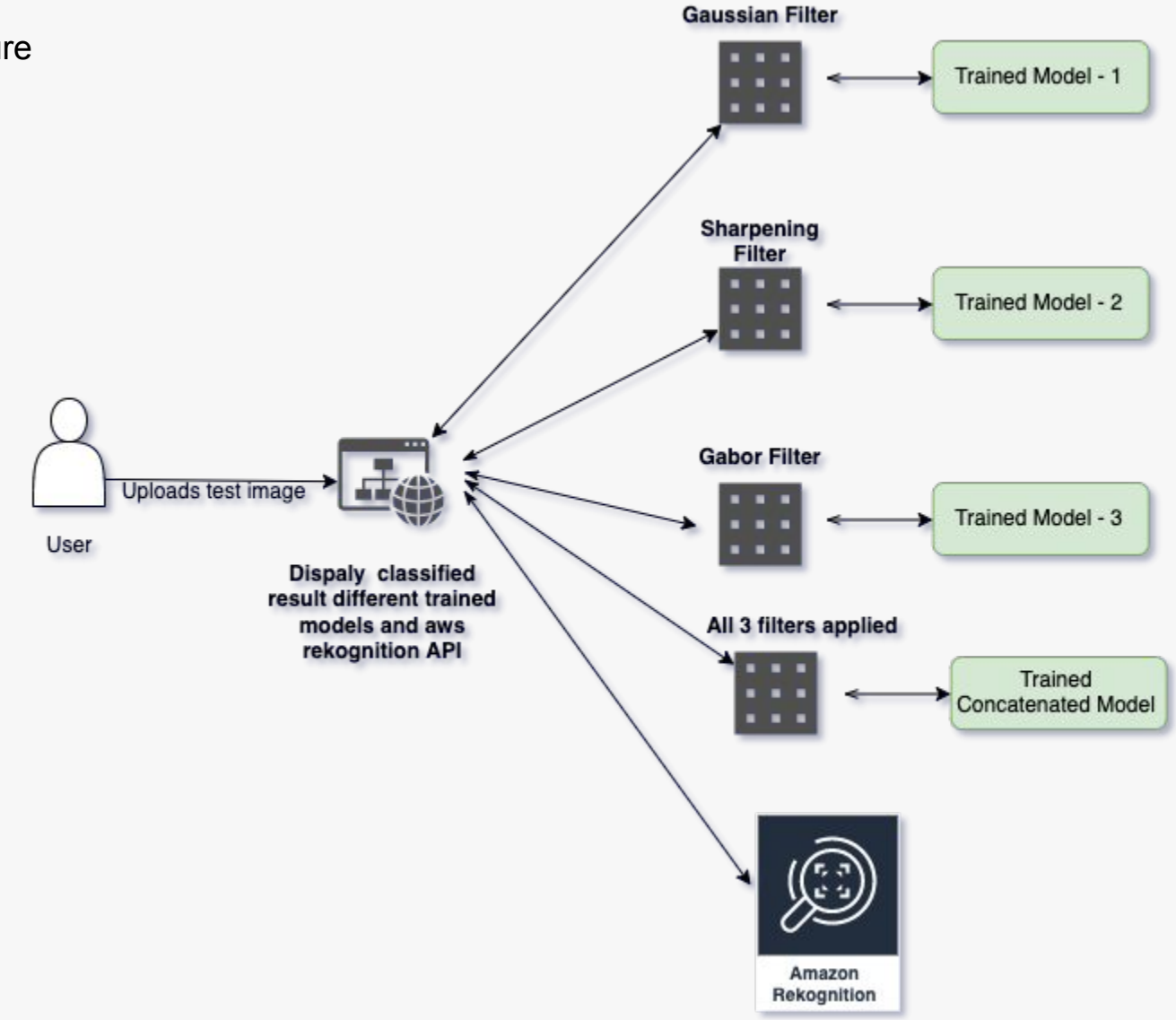
Merged CNN



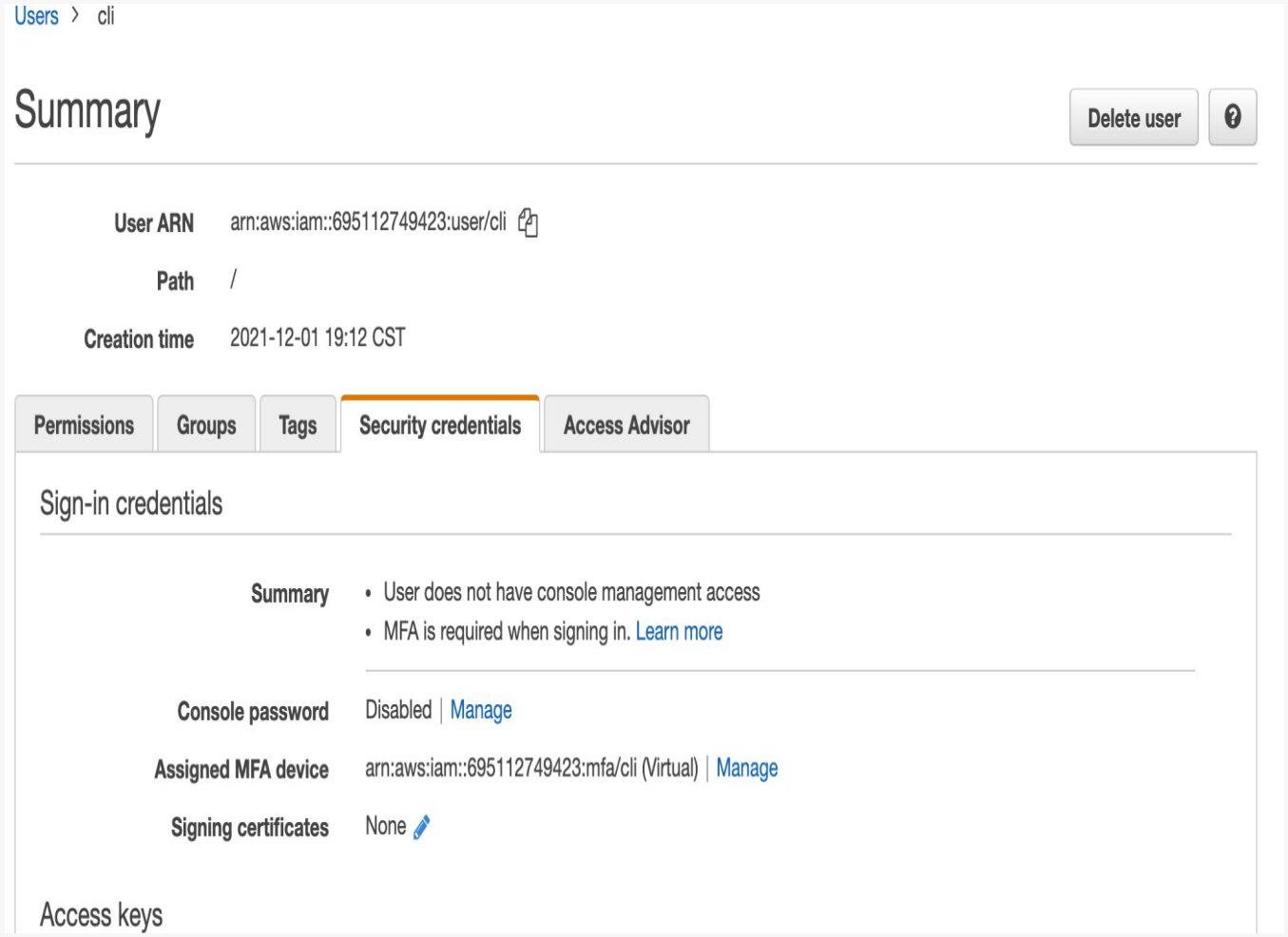
Sharpening CNN

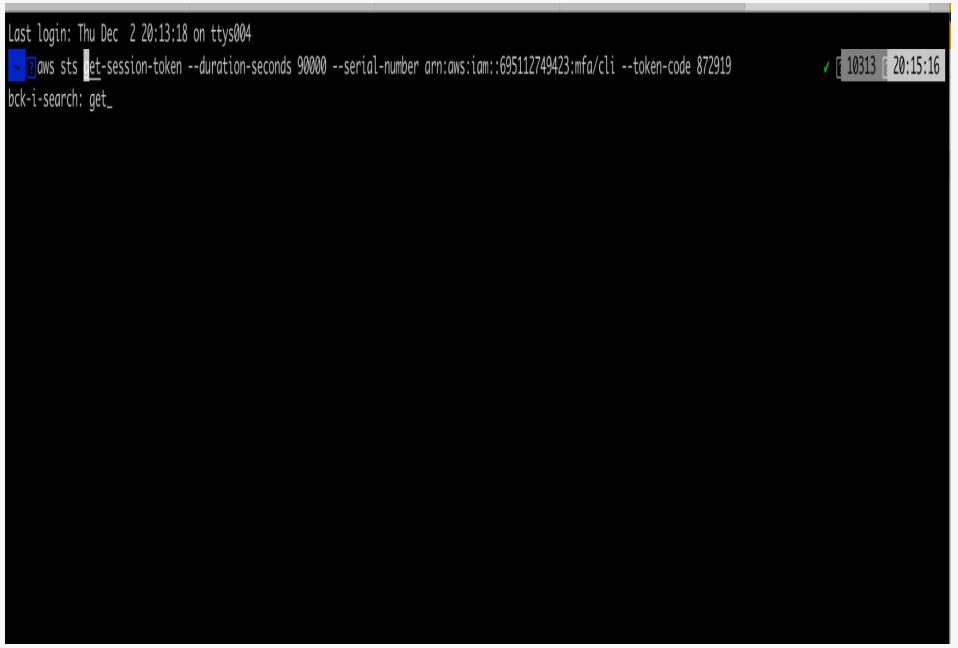
Confusion matrix

Flask Website Architecture



AWS CLI Setup and fetching credentials for Rekognition API





AWS Reckognition API

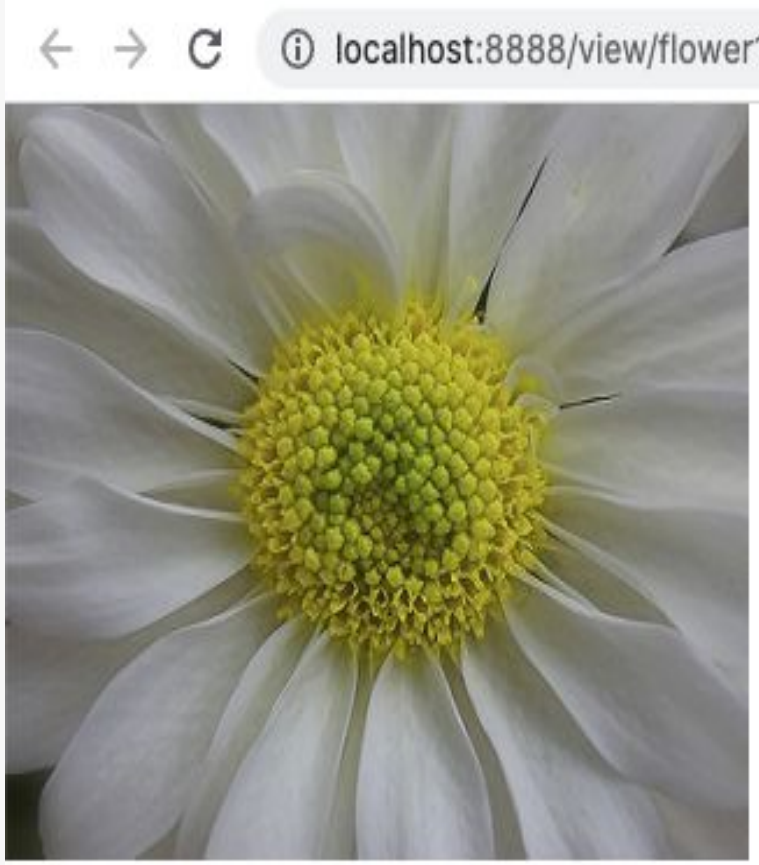
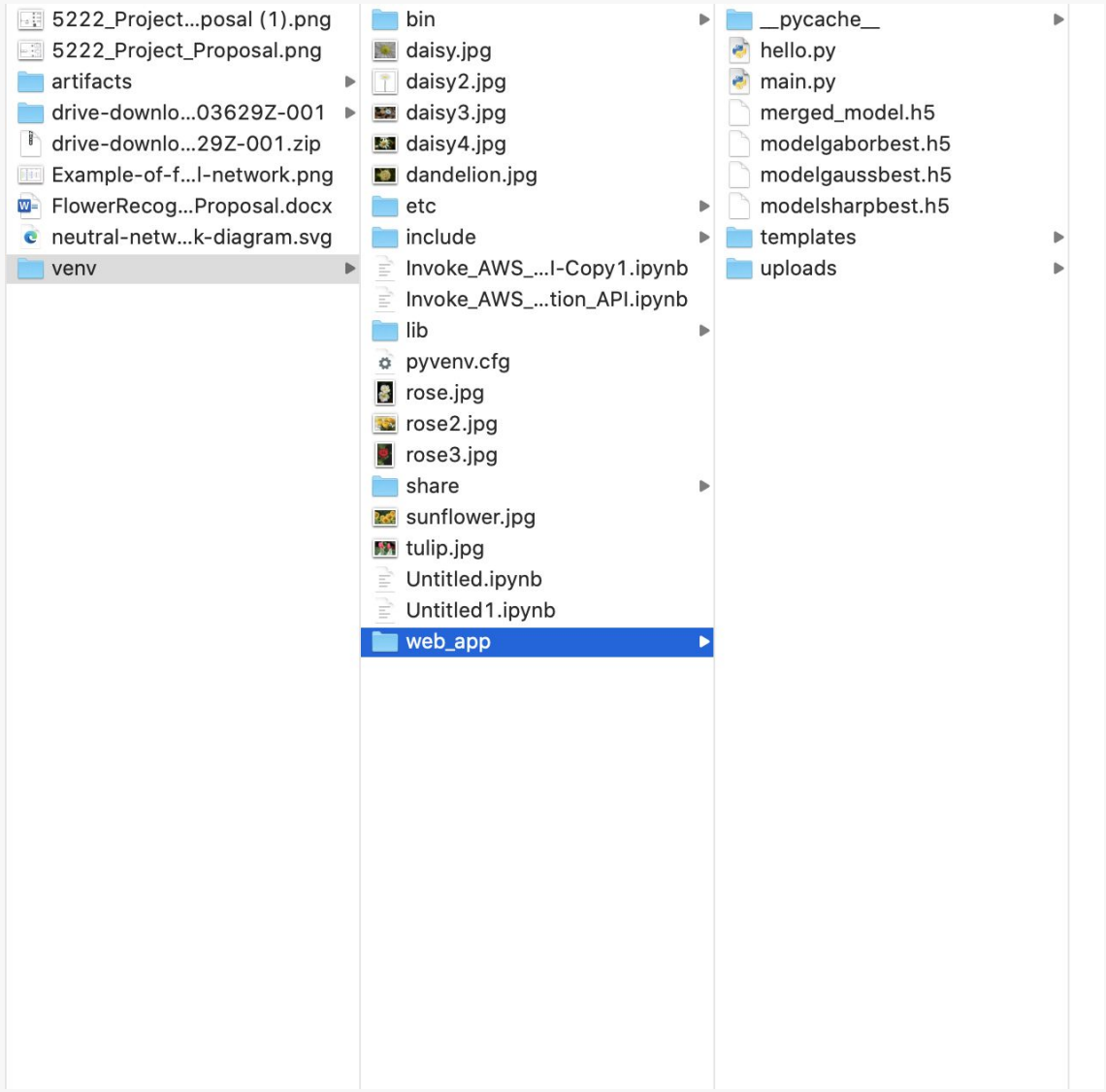


Image given as input to the API

```
In [12]: import boto3

In [13]: # rekognition = boto3.client("rekognition", "us-east-1")

In [14]: rekognition = boto3.client(
             'rekognition',
             aws_access_key_id='ASIAVZKUOEYSO6TRIRN2',
             aws_secret_access_key='0ydsbB2Cj6wtv5K9fZ4pSnRte7BzVU3fZ0y+wF6i',
             aws_session_token='IQoJb3JpZ2luX2VjEFUaCXVzLWVhc3QtMSJGMEQCIEPrdPLWItNjFGmq2Tfbgc1r9H8mHrZWwOE36JYY7uLsAiAvW5QIpsxm
         )

In [15]: import os

         with open(os.path.join('flower1.jpg'), 'rb') as image_data:
                 response_content = image_data.read()
                 rekognition_response = rekognition.detect_labels(Image={'Bytes':response_content})

In [11]: rekognition_response

Out[11]: {'Labels': [{'Name': 'Plant',
            'Confidence': 99.70489501953125,
            'Instances': [{'BoundingBox': {'Width': 0.8668731451034546,
               'Height': 0.8615646958351135,
               'Left': 0.07573075592517853,
               'Top': 0.07832533866167068},
              'Confidence': 59.97138977050781}],
            'Parents': []},
           {'Name': 'Daisy',
            'Confidence': 98.83956146240234,
            'Instances': [],
            'Parents': [{'Name': 'Flower'}, {'Name': 'Plant'}]},
           {'Name': 'Daisies',
            'Confidence': 98.83956146240234,
            'Instances': [],
            'Parents': [{'Name': 'Flower'}, {'Name': 'Plant'}]},
           {'Name': 'Flower',
            'Confidence': 98.83956146240234,
            'Instances': [],
            'Parents': [{'Name': 'Plant'}]},
```

Response from Amazon Rekognition API

Flask Website Setup



Upload, classify - code in flask website

```python
@app.route('/result', methods=['GET'])
def display_result():
    filename = request.args.get('filename')

    # build data frame that store result for image classification
    df_predictions = pd.DataFrame(columns=['Model', 'Predicted Flower Class', 'Probability'])

    # load uploaded image
    uploaded_image = plt.imread(os.path.join(app.config['UPLOAD_FOLDER'], filename))

    # resize image
    resize_img = cv2.resize(uploaded_image, (150,150))

    #convert to np from image
    img_as_np = np.array(resize_img)

    df_predictions, x1 = gaussian_filter_cnn_classify(df_predictions, img_as_np)
    df_predictions, x2 = gabor_filter_cnn_classify(df_predictions, img_as_np)
    df_predictions, x3 = sharpening_filter_cnn_classify(df_predictions, img_as_np)
    df_predictions = merged_model_ccn(df_predictions, [x1, x2, x3])
    df_predictions = aws_rekognition_classify(filename, df_predictions)

    return render_template('result.html', url=filename, predictions=df_predictions)

> def get_flower_name_from_class(class_number):…

> def gaussian_filter_cnn_classify(df_predictions, img_np):…

> def gabor_filter_cnn_classify(df_predictions, img_np):…

> def sharpening_filter_cnn_classify(df_predictions, img_np):…


> def merged_model_ccn(df_predictions, model_input):…


> def aws_rekognition_classify(filename, df_predictions):…
```

Result 1

Uploaded Image

Go Back

Model	Predicted Flower Class	Probability
Gaussian Filter CNN	Dandelion	0.9573889970779419
Gabor Filter CNN	Dandelion	0.9605498909950256
Sharpening Filter CNN	Dandelion	0.9979783892631531
Merged Model CNN	Dandelion	0.9670974612236023
Amazon Rekognition API	Dandelion	0.89904

[redacted]

Result 2

Uploaded Image

Go Back

Model	Predicted Flower Class	Probability
Gaussian Filter CNN	Sunflower	0.988854169845581
Gabor Filter CNN	Sunflower	0.8464449644088745
Sharpening Filter CNN	Sunflower	0.9742432832717896
Merged Model CNN	Sunflower	0.8552935719490051
Amazon Rekognition API	Sunflower	0.960577

Result 3

Uploaded Image

Go Back

Model	Predicted Flower Class	Probability
Gaussian Filter CNN	Sunflower	0.4323742985725403
Gabor Filter CNN	Daisy	0.4821966886520386
Sharpening Filter CNN	Dandelion	0.46889781951904297
Merged Model CNN	Sunflower	0.430497944355011
Amazon Rekognition API	Daisy	0.988396

Result 4

Uploaded Image

Go Back

Model	Predicted Flower Class	Probability
Gaussian Filter CNN	Tulip	0.9948581457138062
Gabor Filter CNN	Tulip	0.989429771900177
Sharpening Filter CNN	Tulip	0.99793940782547
Merged Model CNN	Tulip	0.7826806902885437
Amazon Rekognition API	Tulip	0.94513