

Model Preparation

```
1 # Import libraries
2 import tensorflow as tf
3 import pathlib
4
5 # Define parameters
6 IMG_HEIGHT = 224
7 IMG_WIDTH = 224
8 BATCH_SIZE = 32
9
10 # Download dataset
11 dataset_url =
12     "https://storage.googleapis.com/download.tensorflow.org/example_images/flower_photos.tgz"
13 data_dir = tf.keras.utils.get_file('flower_photos', origin=dataset_url,
14     untar=True)
15 data_dir = pathlib.Path(data_dir)
16
17 # Create train dataset with 20% validation split
18 train_ds = tf.keras.utils.image_dataset_from_directory(
19     data_dir,
20     validation_split=0.2,
21     subset="training",
22     seed=123,
23     image_size=(IMG_HEIGHT, IMG_WIDTH),
24     batch_size=BATCH_SIZE)
25
26 # Create validation dataset with 20% validation split
27 val_ds = tf.keras.utils.image_dataset_from_directory(
28     data_dir,
29     validation_split=0.2,
30     subset="validation",
31     seed=123,
32     image_size=(IMG_HEIGHT, IMG_WIDTH),
33     batch_size=BATCH_SIZE)
34
35 # Get label names and their number
36 class_names = train_ds.class_names
37 num_classes = len(class_names)
38
39 # Configure the dataset for performance
40 AUTOTUNE = tf.data.AUTOTUNE
41
42 train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)
43 val_ds = val_ds.cache().prefetch(buffer_size=AUTOTUNE)
44
45 # Create data augmentation layers
46 data_augmentation = tf.keras.Sequential(
47     [
48         tf.keras.layers.RandomFlip("horizontal",
49             input_shape=(IMG_HEIGHT,
50                 IMG_WIDTH,
51                 3)),
52         tf.keras.layers.RandomRotation(0.1),
53         tf.keras.layers.RandomZoom(0.1),
54     ]
55 )
```

```

54 # Create the model
55 model = tf.keras.Sequential([
56     data_augmentation,
57     tf.keras.layers.Rescaling(1./255),
58     tf.keras.layers.Conv2D(16, 3, padding='same', activation='relu'),
59     tf.keras.layers.MaxPooling2D(),
60     tf.keras.layers.Conv2D(32, 3, padding='same', activation='relu'),
61     tf.keras.layers.MaxPooling2D(),
62     tf.keras.layers.Conv2D(64, 3, padding='same', activation='relu'),
63     tf.keras.layers.MaxPooling2D(),
64     tf.keras.layers.Dropout(0.2),
65     tf.keras.layers.Flatten(),
66     tf.keras.layers.Dense(128, activation='relu'),
67     tf.keras.layers.Dense(num_classes)
68 ])
69
70 # Compile the model
71 model.compile(optimizer='adam',
72               loss=tf.keras.losses.SparseCategoricalCrossentropy(
73                   from_logits=True),
74               metrics=['accuracy'])
75 # Train the model
76 epochs = 15
77 history = model.fit( train_ds,
78                     validation_data=val_ds,
79                     epochs=epochs
80 )
81 # Save the model
82 model.save('model/model.h5')

```

HTML

```

1 <html lang="en">
2
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <meta http-equiv="X-UA-Compatible" content="ie=edge">
7     <title>Flask Deployment</title>
8     <link href=
9         "https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css" rel=
10        "stylesheet">
11     <script src=
12         "https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
13     <script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></
14        script>
15     <script src=
16         "https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
17     <link rel="stylesheet" href=
18         "https://cdn.jsdelivr.net/npm/font-awesome@4.7.0/css/font-awesom
19         e.min.css"
20     >
21     <link href="{{ url_for('static', filename='css/main.css') }}" rel=
22        "stylesheet">
23 </head>

```

```

15
16 <body>
17     <div class="container">
18         <div id="content">{% block content %}{% endblock %}</div>
19     </div>
20 </body>
21
22 <footer>
23     <script src="{ { url_for('static', filename='js/main.js') } }" type=
    "text/javascript"></script>
24 </footer>
25
26 </html>

```

```

1  {% extends "base.html" %} {% block content %}
2
3  <h2>Flower Image Classifier</h2>
4  <div>
5      <form id="upload-file" method="post" enctype="multipart/form-data"
    class=uploader>
6          <input type="file" name="file" id="imageUpload" accept="image/*">
7          <label for="imageUpload" id="file-drag">
8              <div class="image-section" style="display:none;">
9                  <div class="img-preview">
10                     <div id="imagePreview">
11                         </div>
12                     </div>
13                 </div>
14                 <div id="start">
15                     <i class="fa fa-cloud-upload"></i>
16                     <div>
17                         <h5>Select a file or drag here</h5>
18                     </div>
19                     <span id="file-upload-btn" class="btn btn-primary">
    Select a file</span>
20                 </div>
21             </label>
22         </div>
23         <button type="button" class="btn btn-primary btn-lg " id=
    "btn-predict" style="display:none;">Predict</button>
24         <div class="loader" style="display:none;"></div>
25         <h3 id="result">
26             <span> </span>
27         </h3>
28     </div>
29 </form>
30 </div>
31
32 {% endblock %}

```

CSS

```
1  html,
2  body,
3  * {
4      box-sizing: border-box;
5      font-size: 16px;
6  }
7
8  html,
9  body {
10     height: 100%;
11     text-align: center;
12 }
13
14 body {
15     padding: 2rem;
16     background-color: #3b3b3b;
17     color: white;
18 }
19
20 h2,
21 h3 {
22     color: #ef6a15;
23 }
24
25 .container {
26     display: flex;
27     flex-direction: column;
28     justify-content: center;
29     min-height: 90vh;
30 }
31
32 .uploader {
33     display: block;
34     clear: both;
35     margin: 0 auto;
36     width: 100%;
37     max-width: 600px;
38 }
39
40 .uploader div {
41     margin: 0 0 0.5rem 0;
42 }
43
44 .uploader label {
45     float: left;
46     clear: both;
47     width: 100%;
48     padding: 2rem 1.5rem;
49     text-align: center;
50     border-radius: 7px;
51     border: 3px dashed #5c5c5c;
52     transition: all 0.2s ease;
53     -webkit-user-select: none;
54     -moz-user-select: none;
55     -ms-user-select: none;
56     user-select: none;
57 }
58
59 .uploader label:hover {
60     border-color: #ef6a15;
61 }
62
63 .uploader label:hover {
64     border: 3px solid #ef6a15;
65 }
66
67 .uploader label:hover #start i.fa {
68     transform: scale(0.8);
69     opacity: 0.3;
70 }
71
72 .uploader #start {
73     float: left;
74     clear: both;
75     width: 100%;
76 }
```

```

78 .uploader #start.hidden {
79     display: none;
80 }
81
82 .uploader #start i.fa {
83     font-size: 50px;
84     margin-bottom: 1rem;
85     transition: all 0.2s ease-in-out;
86     color: #ef6a15;
87 }
88
89 .img-preview {
90     display: flex;
91     justify-content: center;
92 }
93
94 .img-preview>div {
95     width: 224px;
96     height: 224px;
97     background-size: cover;
98     background-repeat: no-repeat;
99     background-position: center;
100 }
101
102 .uploader #notimage {
103     display: block;
104     float: left;
105     clear: both;
106     width: 100%;
107 }
108
109 input[type="file"] {
110     display: none;
111 }
112
113 .uploader .btn {
114     display: inline-block;
115     margin: 0.5rem 0.5rem 1rem 0.5rem;
116     clear: both;
117     font-family: inherit;
118     font-weight: 700;
119     font-size: 14px;
120     text-decoration: none;
121     text-transform: initial;
122     border: none;
123     border-radius: 0.2rem;
124     outline: none;
125     padding: 0 1rem;
126     height: 36px;
127     line-height: 36px;
128     color: #fff;
129     transition: all 0.2s ease-in-out;
130     box-sizing: border-box;
131     background: #ef6a15;
132     border-color: #ef6a15;
133     cursor: pointer;
134 }
135
136 .loader {
137     display: inline-block;
138     border: 8px solid #f3f3f3;
139     border-top: 8px solid #ef6a15;
140     border-radius: 50%;
141     width: 50px;
142     height: 50px;
143     animation: spin 1s linear infinite;
144 }
145
146 @keyframes spin {
147     0% {
148         transform: rotate(0deg);
149     }
150     100% {
151         transform: rotate(360deg);
152     }
153 }

```


JavaScript

```
1 $(document).ready(function() {
2     // Init
3     $('.image-section').hide();
4     $('.loader').hide();
5     $('#result').hide();
6     var fileSelect = document.getElementById('imageUpload'),
7         fileDrag = document.getElementById('file-drag');
8     fileSelect.addEventListener('change', fileSelectHandler, false);
9
10    // File Drop
11    fileDrag.addEventListener('dragover', fileDragHover, false);
12    fileDrag.addEventListener('dragleave', fileDragHover, false);
13    fileDrag.addEventListener('drop', fileSelectHandler, false);
14
15    function fileDragHover(e) {
16        var fileDrag = document.getElementById('file-drag');
17        e.stopPropagation();
18        e.preventDefault();
19
20        fileDrag.className = (e.type === 'dragover' ? 'hover' : 'modal-body imageUpload');
21    }
22
23    function fileSelectHandler(e) {
24        // Fetch FileList object
25        var files = e.target.files || e.dataTransfer.files;
26
27        // Cancel event and hover styling
28        fileDragHover(e);
29        e.preventDefault();
30        fileInput = document.getElementById("imageUpload");
31        fileInput.files = files;
32        readURL(files[0]);
33        console.log("changed")
34        $('.image-section').show();
35        $('#btn-predict').show();
36        $('#result').text('');
37        $('#result').hide();
38    }
39
40    // Upload Preview
41    function readURL(input) {
42        if (input) {
43            var reader = new FileReader();
44            reader.onload = function(e) {
45                $('#imagePreview').css('background-image', 'url(' + e.target.result + ')');
46            }
47            if (input.files && input.files[0]) {
48                reader.readAsDataURL(input.files[0]);
49                console.log("input 0:" + input.files[0]);
50            } else {
51                reader.readAsDataURL(input);
52                console.log("input 1:" + input);
53            }
54        }
55    }
```

```
57    // Predict
58    $('#btn-predict').click(function() {
59        var form_data = new FormData($('#upload-file')[0]);
60        // Show loading animation
61        $(this).hide();
62        $('.loader').show();
63        // Make prediction by calling api /predict
64        $.ajax({
65            type: 'POST',
66            url: '/predict',
67            data: form_data,
68            contentType: false,
69            cache: false,
70            processData: false,
71            async: true,
```

```

72         success: function(data) {
73             // Get and display the result
74             $('#loader').hide();
75             $('#result').fadeIn(600);
76             $('#result').text(' Result: ' + data);
77             console.log('Success!');
78         },
79     });
80 });
81
82 });

```

Main App

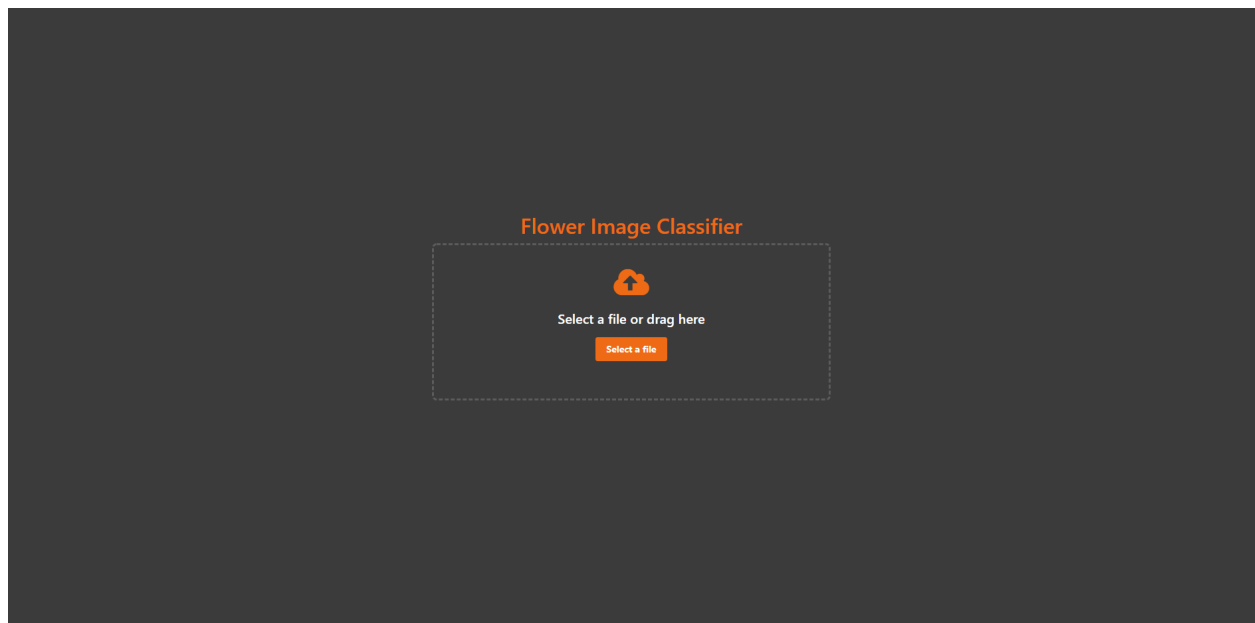
```

1  import os
2  import numpy as np
3  import tensorflow as tf
4  from flask import Flask, request, render_template
5  from werkzeug.utils import secure_filename
6
7  # Define a flask app
8  app = Flask(__name__)
9  app.config['SEND_FILE_MAX_AGE_DEFAULT'] = -1
10 MODEL_PATH = 'model/model.h5'
11
12 # Load your trained model
13 model = tf.keras.models.load_model(MODEL_PATH)
14
15 IMG_HEIGHT = 224
16 IMG_WIDTH = 224
17 class_names = ['Daisy', 'Dandelion', 'Roses', 'Sunflowers', 'Tulips']
18
19 # Predict
20 def model_predict(img_path,model):
21
22     img = tf.keras.utils.load_img(img_path, target_size=(IMG_HEIGHT,
23     IMG_WIDTH))
24     img_array = tf.keras.utils.img_to_array(img)
25     img_array = tf.expand_dims(img_array, 0) # Create a batch
26
27     preds = model.predict(img_array)
28     return preds
29
30 # Decode the prediction of the model
31 def decode_prediction(predictions):
32     score = tf.nn.softmax(predictions[0])
33     percentage_acc = 100 * np.max(score)
34     return "{} with {:.2f}% Confidence".format(class_names[np.argmax(score
35     )],percentage_acc)
36
37 @app.route('/', methods=['GET'])
38 def index():
39     # Main page
40     return render_template('index.html')

```

```
41 @app.route('/predict', methods=['GET', 'POST'])
42 def upload():
43     if request.method == 'POST':
44         # Get the file from post request
45         f = request.files['file']
46
47         # Save the file to ./uploads
48         basepath = os.path.dirname(__file__)
49         file_path = os.path.join(
50             basepath, 'uploads', secure_filename(f.filename))
51         f.save(file_path)
52
53         # Make prediction
54         preds = model_predict(file_path, model)
55
56         # Process result
57         result = decode_prediction(preds)
58         return result
59     return None
60
61
62 if __name__ == '__main__':
63     app.run(debug=True)
```

Demo



Flower Image Classifier



Select a file or drag here



Select a file

Flower Image Classifier

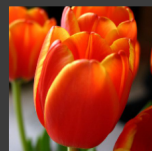


Select a file or drag here

Select a file

Predict

Flower Image Classifier

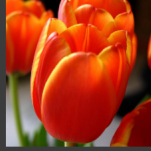


Select a file or drag here

Select a file



Flower Image Classifier



Select a file or drag here

Select a file

Result: Tulips with 98.18% Confidence