Sanika Patil

Roll No. 46

Virtual Image Search Engine

Build_Features.py

```
import os
import pickle
import numpy as np
from tensorflow.keras.applications.vgg16 import VGG16, preprocess_input
from tensorflow.keras.preprocessing import image
DATASET_PATH = "static/dataset"
os.makedirs(DATASET_PATH, exist_ok=True)  # dataset folder तयार करतो
# Pretrained VGG16 model
model = VGG16(weights='imagenet', include_top=False, pooling='avg')
filenames = []
features_list = []
for img_name in os.listdir(DATASET_PATH):
    img_path = os.path.join(DATASET_PATH, img_name)
    try:
        img = image.load_img(img_path, target_size=(224,224))
        x = image.img_to_array(img)
        x = np.expand_dims(x, axis=0)
        x = preprocess_input(x)
        features = model.predict(x).flatten()
        filenames.append(img_name)
        features_list.append(features)
    except Exception as e:
        print(f"Error processing {img_name}: {e}")
```

```
features_array = np.array(features_list)
with open("features.pkl", "wb") as f:
    pickle.dump((filenames, features_array), f)
print("Features saved successfully!")
```

app.py

```
# app.py
from flask import Flask, render_template, request
import pickle
import numpy as np
import os
from tensorflow.keras.applications.vgg16 import VGG16, preprocess_input
from tensorflow.keras.preprocessing import image
from sklearn.metrics.pairwise import cosine_similarity
app = Flask(__name__)
# Load features and filenames
with open("features.pkl", "rb") as f:
    filenames, features = pickle.load(f)
# Folder for uploaded query images
UPLOAD_FOLDER = "static/uploads"
if not os.path.exists(UPLOAD_FOLDER):
    os.makedirs(UPLOAD_FOLDER)
# Pretrained VGG16 model for feature extraction
model = VGG16(weights='imagenet', include_top=False, pooling='avg')
# Function to extract features of an image
def extract_features(img_path):
    img = image.load_img(img_path, target_size=(224,224))
    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)
```

```
    x = preprocess_input(x)
    feat = model.predict(x)
    return feat.flatten()
@app.route('/')
def home():
    return render_template('index.html')
@app.route('/search', methods=['POST'])
def search():
    # Save uploaded file
    file = request.files['query_img']
    file_path = os.path.join(UPLOAD_FOLDER, file.filename)
    file.save(file_path
    # Extract features of uploaded image
    query_features = extract_features(file_path)
    # Compute similarity with dataset features
    sims = cosine_similarity([query_features], features)[0]
    # Top 3 similar images (exclude uploaded query image)
    top_indices = np.argsort(sims)[::-1]
    top_results = []
    for i in top_indices:
        if filenames[i] != file.filename:  # exclude uploaded image if same filename exists
            top_results.append(filenames[i])
        if len(top_results) == 5:  # get top 3
            break
    # Prepare full path for result images
    results = [os.path.join("static/dataset", img) for img in top_results]
    return render_template('result.html', query=os.path.join("static/uploads", file.filename),
results=results)
if __name__ == '__main__':
    app.run(debug=True)
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Visual Search Engine</title>
  <style>
    @import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap');
    body {
      font-family: 'Poppins', sans-serif;
      display: flex;
      flex-direction: column;
      align-items: center;
      justify-content: center;
      height: 100vh;
      margin: 0;
      background: linear-gradient(-45deg, #ff9a9e, #fad0c4, #a18cd1, #fbc2eb);
      background-size: 400% 400%;
      animation: gradientBG 15s ease infinite;
    }
    @keyframes gradientBG {
      0% {background-position: 0% 50%;}
      50% {background-position: 100% 50%;}
      100% {background-position: 0% 50%;}
    }
```

```
h1 {
    color: #fff;
    font-size: 3rem;
    margin-bottom: 50px;
    text-shadow: 2px 2px 15px rgba(0,0,0,0.3);
}

.upload-box {
    background: rgba(255, 255, 255, 0.95);
    border-radius: 25px;
    padding: 50px 60px;
    box-shadow: 0px 10px 40px rgba(0,0,0,0.35);
    transition: transform 0.3s, box-shadow 0.3s;
}
.upload-box:hover {
    transform: translateY(-5px);
    box-shadow: 0px 15px 50px rgba(0,0,0,0.4);
}
input[type=file] {
    margin: 20px 0;
    padding: 12px;
    border-radius: 12px;
    border: 1px solid #ccc;
    cursor: pointer;
    font-size: 1rem;
}
button {
    background: linear-gradient(90deg, #ff6a00, #ee0979);
    color: white;
```

```
            border: none;
            padding: 14px 35px;
            font-size: 1.1rem;
            font-weight: 600;
            border-radius: 15px;
            cursor: pointer;
            transition: background 0.3s, transform 0.3s;
        }
        button:hover {
            background: linear-gradient(90deg, #ee0979, #ff6a00);
            transform: scale(1.05);
        }
        @media (max-width: 600px) {
            h1 {
                font-size: 2rem;
                margin-bottom: 30px;
            }
            .upload-box {
                padding: 30px 20px;
            }
            button {
                padding: 12px 25px;
            }
        }
    </style>
</head>
<body>
    <h1> Visual Search Engine</h1>
    <div class="upload-box">
```

```
    <form action="/search" method="POST" enctype="multipart/form-data">
      <input type="file" name="query_img" required>
      <br>
      <button type="submit">Search Similar Images</button>
    </form>
  </div>
</body>
</html>
```

Result.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Top Similar Images</title>
  <style>
    body {
      background: linear-gradient(135deg, #f0f4f8, #d9e2ec);
      font-family: "Poppins", sans-serif;
      text-align: center;
      padding: 60px 20px;
    }

    h1 {
      color: #1f2937;
      font-size: 3rem;
      margin-bottom: 40px;
      text-shadow: 2px 2px 10px rgba(0,0,0,0.2);
```

```
}

h3 {
    color: #1f2937;
    margin-bottom: 25px;
    font-size: 1.5rem;
}

.image-grid {
    display: flex;
    flex-wrap: wrap;
    justify-content: center;
    gap: 30px;
}

.image-card {
    position: relative;
    overflow: hidden;
    border-radius: 25px;
    width: 280px;
    height: 280px;
    box-shadow: 0px 12px 30px rgba(0,0,0,0.25);
    transition: transform 0.4s, box-shadow 0.4s, border 0.4s;
    border: 3px solid transparent;
    background: #fff;
}

.image-card img {
    width: 100%;
```

```
    height: 100%;
    object-fit: cover;
    display: block;
    border-radius: 25px;
}

.image-card:hover {
    transform: scale(1.1);
    box-shadow: 0px 18px 40px rgba(0,0,0,0.35);
    border: 3px solid #ff6a00;
}

.back-btn {
    display: inline-block;
    margin-top: 50px;
    padding: 15px 40px;
    background: linear-gradient(90deg, #ff6a00, #ee0979);
    color: white;
    text-decoration: none;
    font-weight: 600;
    font-size: 1.2rem;
    border-radius: 20px;
    transition: background 0.3s, transform 0.3s;
}

.back-btn:hover {
    background: linear-gradient(90deg, #ee0979, #ff6a00);
    transform: scale(1.05);
}
```

```
    </style>
</head>
<body>
    <h1>Top Similar Images</h1>
    <h3>Query Image</h3>
    <div class="image-grid">
        <div class="image-card">
            <img src="/{{ query|replace('\\','/') }}" alt="Query Image">
        </div>
    </div>

    <h3>Similar Images</h3>
    <div class="image-grid">
        {% for r in results %}
            <div class="image-card">
                <img src="/{{ r|replace('\\','/') }}" alt="Similar Image">
            </div>
        {% endfor %}
    </div>

    <a href="/" class="back-btn">⬅ Back to Search</a>
</body>
</html>
```