Project_2 readme

Name: Priyanshu Ranka

NUID: 002305396

Course and Term: PRCV, Spring 2025

OS and IDE: Windows 11, Visual Studio

NO TIME TRAVEL DAYS ARE BEING USED.

Content-Based Image Retrieval (CBIR) Using Classical and Deep Learning Features

Overview

This project implements **Content-Based Image Retrieval (CBIR)** using a combination of **classical feature extraction techniques** and **deep learning-based embeddings**. The goal is to retrieve the most visually similar images to a target image from a dataset by computing feature similarities.

Implemented Methods

The project explores multiple image retrieval approaches:

1. **Baseline Matching (Central Patch & SSD)**
    - Uses a **7×7 patch** from the center of each image.
    - Computes **Sum of Squared Differences (SSD)** for similarity measurement.
2. **Histogram-Based Matching**
    - Extracts **2D color histograms (rg chromaticity)** from images.
    - Uses **Histogram Intersection** to measure similarity.
3. **Multi-Histogram Matching**
    - Extracts **multiple histograms from different image regions**.
    - Combines regional histograms using a **weighted distance metric**.
4. **Texture and Color-Based Matching**
    - Extracts **whole-image color histograms**.
    - Computes **Sobel-based texture histograms**.
    - Uses a **combined metric (equal weighting of color and texture)**.
5. **Deep Network Embeddings (ResNet18)**
    - Uses **precomputed 512-dimensional feature vectors** from a **ResNet18 model trained on ImageNet**.
    - Computes **SSD distance** between embeddings for similarity matching.

6. **Comparison of Deep Learning vs. Classical Features**
    - Evaluates retrieval performance of **ResNet18 embeddings** vs. **color and texture-based features**.
7. **Custom CBIR System (For Category-Specific Retrieval - Shoes)**
    - Designs a **domain-specific feature vector** for retrieving images of **shoes**.

Project Structure

```
📂 Project_2_Submission
├── 📂 code/              # C++ code implementation
├── 📂 data/              # Dataset and feature files
├── 📂 results/           # Output images & retrieval results
├── 📂 olympus/           # Image directory
├── 📄 ResNet18_olym.csv  # Precomputed ResNet18 embeddings
├── 📄 report.pdf         # Detailed project report
├── 📄 README.md          # Project documentation (this file)
```

Setup & Dependencies

1. Requirements

- **C++** (GCC or MSVC)
- **OpenCV** (for image loading & visualization)
- **CMake** (for compiling the project)

2. Installation

```
sudo apt-get install libopencv-dev  # For Ubuntu
brew install opencv                 # For macOS
```

3. Compiling the Code

```
g++ -o image_retrieval main.cpp `pkg-config --cflags --libs opencv4`
```

Usage

1. Running Baseline Matching

./image_retrieval pic.0893.jpg 3

- The program retrieves the **top 3 closest matches** for pic.0893.jpg.

2. Running ResNet18-Based Retrieval

./image_retrieval pic.0164.jpg 5

- The program loads **ResNet18 feature vectors** and finds the **top 5 matches**.

Acknowledgements

This project was completed as part of the **Pattern Recognition and Computer Vision (PRCV) course** at **Northeastern University**. Special thanks to:

- **Professor Bruce Maxwell** for valuable insights.
- **Teaching Assistants (TAs)** for their guidance.
- **OpenCV Documentation & GitHub Resources** for helpful implementations.

Future Improvements

- Implement **Cosine Distance** for ResNet18 embeddings.
- Explore **additional texture descriptors (GLCM, Gabor filters, etc.)**.
- Optimize performance using **KD-Trees or Approximate Nearest Neighbors (ANNs)**.