MACHINE LEARNING

COMPLEX COMPUTING ACTIVITY

Image Clustering and Retrieval System

PROJECT REPORT

GROUP PARTICIPANTS

Sumama Suleman Madda (22F-BSAI-12)

Muhammad Kaif Abdullah (22F-BSAI-19)

Muzna Siddiqui (22F-BSAI-37)

Maheen Siddiqui (22F-BSAI-49)

OBJECTIVE

Design and implement a complex, multi-faceted ML project that integrates a wide range of concepts from the labs. The goal is to demonstrate end-to-end mastery: data collection/preprocessing, model design, evaluation, deployment/visualization (or real-time operation), and interpretation.

PROJECT DESCRIPTION

This is your final, open-ended project. You are expected to design and build a significant ML application that is more advanced than any single lab. Choose a domain (e.g., healthcare triage, retail sales forecasting, autonomous vehicle assist, surveillance analytics, AR educational tool) and integrate model-building with system aspects (real-time pipeline, visualization, lightweight deployment, or AR overlay).

Image Clustering and Retrieval System

Introduction

The rapid growth of digital image data has created a need for efficient systems that can organize and retrieve images without relying on manual annotations. Traditional text-based image retrieval approaches are limited by incomplete or inaccurate labeling. To overcome these limitations, this project implements a **Content-Based Image Retrieval (CBIR) and Image Clustering System** that operates directly on visual content.

The system combines **deep learning–based feature extraction** with classical machine learning algorithms to demonstrate an end-to-end ML workflow. A pretrained ResNet50 model is used to extract semantic image features, which are then clustered using unsupervised techniques and optionally classified using a Support Vector Machine (SVM). The project integrates multiple lab concepts into a single, practical application.

System Architecture and Pipeline

The system follows a modular pipeline in which each stage performs a specific task and feeds into the next. This design improves clarity, extensibility, and debugging.

Images (ZIP) → Preprocessing → ResNet50 Feature Extraction → Feature Scaling → K-Means / Hierarchical Clustering → Evaluation & Visualization → SVM (optional) → Image Retrieval UI

Images are first preprocessed and converted into deep feature embeddings. These embeddings are clustered, evaluated, and used for similarity-based image retrieval through an interactive interface.

Integration of Lab Concepts

This project explicitly integrates the following lab concepts:

- **Data Preprocessing:** Image resizing, RGB conversion, dataset balancing, and feature scaling using StandardScaler.
- **K-Means Clustering:** Used as the primary unsupervised method to group visually similar images.
- **Hierarchical Agglomerative Clustering:** Applied to analyze hierarchical relationships between images.
- **Support Vector Machine (SVM):** Trained on a labeled subset of embeddings for supervised image classification.

Dataset Description

The dataset is provided as a compressed archive containing images organized into class-wise folders. Each folder represents a semantic category. Images vary in resolution, background, and lighting conditions.

- Formats: JPG/PNG
- Color space: RGB
- Labels: Folder-based (used only for SVM training and evaluation)

Data Preprocessing

Preprocessing ensures consistency and reliable feature extraction. Images are resized to **224×224 pixels** to match ResNet50 input requirements, converted to RGB format, and cleaned by skipping corrupted files. Dataset balancing is applied by limiting the number of images per class. Images are then converted into NumPy arrays for efficient processing.

Feature Extraction and Scaling

A pretrained **ResNet50** model (ImageNet weights) is used as a fixed feature extractor. The final classification layers are removed, and Global Average Pooling produces a **2048-dimensional embedding** for each image. These embeddings capture high-level semantic information.

The embeddings are normalized using **StandardScaler**, which is essential for distance-based algorithms such as K-Means, Hierarchical Clustering, and SVM.

For visualization only, **Principal Component Analysis (PCA)** is applied to project embeddings into two dimensions.

Model Design

K-Means Clustering

K-Means partitions image embeddings into a predefined number of clusters based on Euclidean distance. The algorithm uses k-means++ initialization and a maximum of 300 iterations. It is efficient and well-suited for large datasets.

Hierarchical Agglomerative Clustering

Agglomerative clustering with Ward linkage is used to study hierarchical similarities between images. Unlike K-Means, it does not require predefining the number of clusters and provides insight into multi-level relationships.

Support Vector Machine

An SVM with an RBF kernel is trained on labeled embeddings. With parameters **C = 1.0** and **gamma = scale**, the SVM performs effective classification in high-dimensional feature space without retraining the deep model.

Evaluation Metrics

- **Silhouette Score:** Measures cluster cohesion and separation.
- **Davies–Bouldin Index:** Evaluates cluster compactness (lower is better).
- **Classification Metrics:** Accuracy, precision, recall, F1-score, and confusion matrix for SVM.

Image Retrieval System

For image retrieval, a query image is converted into a ResNet50 embedding and compared with stored embeddings using **cosine similarity**. The system returns the top-K most similar images. For large-scale deployment, approximate nearest neighbor methods (e.g., FAISS) can be used to improve efficiency.

Challenges, Limitations, and Ethics

Challenges include selecting the optimal number of clusters and handling high-dimensional embeddings. The system relies on pretrained models, which may introduce dataset bias. Ethical deployment requires dataset auditing and cautious use in sensitive applications.

Setup and Execution

Environment Setup

```
conda create -n img_cluster python=3.9
conda activate img_cluster
pip install tensorflow scikit-learn numpy pillow matplotlib streamlit
```

Hardware: CPU (minimum), GPU recommended for faster feature extraction.

Execution:

```
python feature_extraction.py
python clustering.py
python evaluation.py
streamlit run app.py
```

Conclusion

This project demonstrates an end-to-end machine learning system that integrates deep feature extraction, unsupervised clustering, supervised classification, and content-based image retrieval. By combining multiple lab concepts into a unified application, the system showcases both theoretical understanding and practical implementation, making it suitable for real-world image analysis tasks.