MACHINE LEARNING

Complex Computing Activity

PROJECT REPORT

Image Clustering and Retrieval System

Team Members

Abdul Basit (22F-BSAI-25)

Shehryar Ahmed (22F-BSAI-28)

Muhammad Hamamd (22F-BSAI-39)

Muhammad Sohaib (22F-BSAI-40)

Objective: To extract discriminative image features using a pretrained deep CNN.
To compare clustering and classification algorithms for image organization and prediction.
To build an interactive image retrieval system with measurable performance metrics.

Project Description:

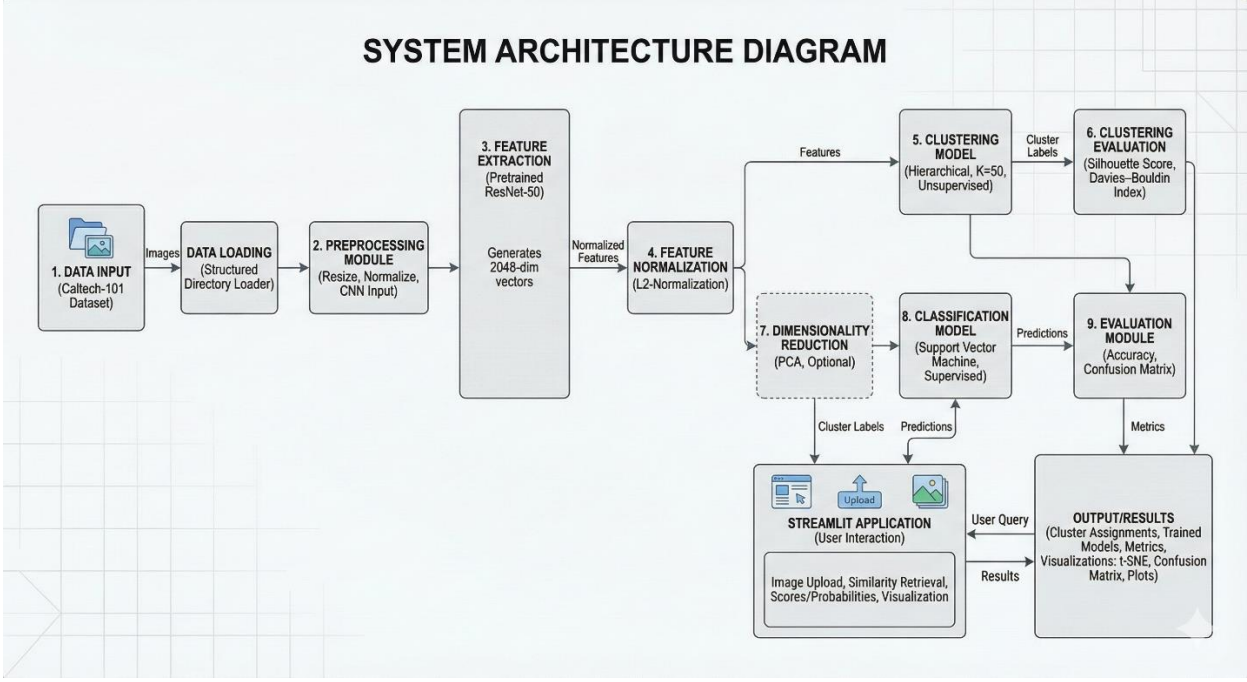
- This project implements an image clustering, retrieval, and classification system using deep feature embeddings.
- Images from the Caltech-101 dataset are processed through a pretrained ResNet-50 model to extract high-level visual features.
- Unsupervised learning techniques (K-Means and Hierarchical clustering) group visually similar images.
- Supervised classifiers (SVM and Random Forest) are trained on extracted features for category prediction.
- A Streamlit application enables users to upload an image and retrieve visually similar images with confidence scores.

1.Introduction:

- With the rapid growth of visual data, automatic image understanding has become essential.
- Traditional image processing methods struggle to capture high-level semantic information.
- Deep learning–based feature extraction combined with classical machine learning provides a practical solution.
- This project demonstrates an end-to-end pipeline for image clustering, classification, and retrieval using real datasets.

2. Architecture & Pipeline

2.1 System Architecture



Components:

1. **Data Input:** Images from the Caltech-101 dataset are loaded from disk using a structured directory-based dataset loader.
2. **Preprocessing Module:** Images are resized, normalized, and passed through a pretrained CNN to extract deep feature embeddings.
3. **Feature Extraction:** A pretrained ResNet-50 model generates 2048-dimensional feature vectors for each image.
4. **Feature Normalization:** Extracted features are L2-normalized to ensure distance-based algorithms behave correctly.
5. **Clustering Model:** Hirarchical (K=50) groups images into visually similar clusters in an unsupervised manner.
6. **Clustering Evaluation:** Silhouette Score and Davies–Bouldin Index are used to assess cluster quality.

7. **Dimensionality Reduction:** Optional PCA reduces feature dimensions before classification to improve efficiency.
8. **Classification Model:** A Support Vector Machine classifier is trained on the extracted features for supervised category prediction.
9. **Evaluation Module:** Classification accuracy and confusion matrices measure model performance.
10. **Output/Results:** The system produces cluster assignments, trained clustering and classification models, evaluation metrics, and visualizations (t-SNE, confusion matrix, performance plots). A Streamlit application enables user interaction by allowing image upload, retrieving visually similar images from the dataset, and displaying similarity scores/probabilities with image names in real time.

2.2 Pipeline Diagram

```
[Caltech-101 Images]
        ↓
[Image Loading]
        ↓
[Resize + Normalize]
        ↓
[ResNet-50 Feature Extraction]
        ↓
[2048-D Feature Vectors]
        ↓
[L2 Normalization]
        ↓
[Hierarchical Agglomerative Clustering (K=50)]
        ↓
[Silhouette & Davies-Bouldin Evaluation]
                ↓
[SVM Classification]
        ↓
[Accuracy + Confusion Matrix]
        ↓
[Results: Clusters, Models, Metrics, t-SNE Plots, image retrieval]
```

3. Lab Concepts Integration

Lab	Lab Concept	How It Was Used	Where in Code / Pipeline
5	Random Forest Classifier	Used as a supervised image classification model trained on ResNet-50 feature embeddings to predict image categories and compared against SVM.	`models/classifier/random_forest.pkl`, training in **Phase 4: Classification**, results in `classification_results.json`, visualization in `results/classification/`
6	Support Vector Machine (SVM) Classifier	Trained as an alternative supervised classifier on the same deep features to compare performance with Random Forest.	`models/classifier/svm_rbf.pkl`, results in `classification_results_svm.json`, accuracy plot `rf_vs_svm_accuracy.png`
10	K-Means Clustering	Applied for unsupervised grouping of image embeddings to analyze visual similarity and cluster structure before retrieval.	`models/clustering/kmeans_k20.pkl`, labels in `cluster_labels.npy`, metrics in `clustering_metrics.json`, elbow plot in `results/clustering/elbow_curve.png`
11	Hierarchical Agglomerative Clustering	Used to create fine-grained visual clusters (K=50) for better semantic grouping and cluster evaluation.	`models/clustering/hir_model_k50.pkl`, labels in `hir_labels_k50.npy`, metrics in `hir_metrics_k50.json`, dendrogram/TSNE in `results/clustering/`

4. Dataset

4.1 Dataset Description

1. **Name:** Caltech-101
2. **Source:** Caltech Vision Lab (official dataset)Size: [Number of samples, file size]
3. **Structure:** ~5,163 images across 51 object categories
4. **Split:** 80% training / 20% validation (for ML)

4.2 Preprocessing Steps

1. **Image Resizing:** All images resized to **224×224** to match CNN input requirements
2. **Normalization:** Pixel values normalized using ImageNet mean and standard deviation
3. **Feature Extraction:** Images passed through pretrained **ResNet-50** to obtain deep embeddings
4. **Feature Normalization:** L2 normalization applied to extracted feature vectors
5. **Dimensionality Reduction:** Optional **PCA** applied before classification for efficiency

5. Model Design & Training

5.1 Model Architecture

Model Type: ResNet50 (Pretrained Model)

Architecture Details:

- **Input Layer:** RGB image, 224×224×3
- **Initial Conv Layer:** 7×7 convolution, 64 filters, stride 2
- **Residual Blocks:**
  - Conv2_x → Conv5_x (bottleneck blocks with skip connections)
- **Global Average Pooling:** Reduces spatial dimensions
- **Output Feature Layer:** 2048-dimensional feature vector
- **Classification Layer: Removed / not used** (model used as feature extractor only)

5.2 Hyperparameters

Since ResNet-50 is frozen**,** most training hyperparameters do not apply**.**

Hyperparameter	Value	Rationale
Learning Rate	N/A	No CNN fine-tuning performed
Batch Size	32	Efficient GPU/CPU feature extraction
Epochs	1	Single forward pass for embeddings
Optimizer	N/A	Model weights not updated
Loss Function	N/A	No CNN training

5.3 Training Details

- **Hardware Used:** CPU (local machine / laptop)
- **Training Time:** Not applicable (only feature extraction)

- **Framework:** PyTorch

- **Key Techniques:**
  - Transfer learning (feature extraction)
  - L2 feature normalization
  - PCA before SVM/RF classification

6. Evaluation & Results

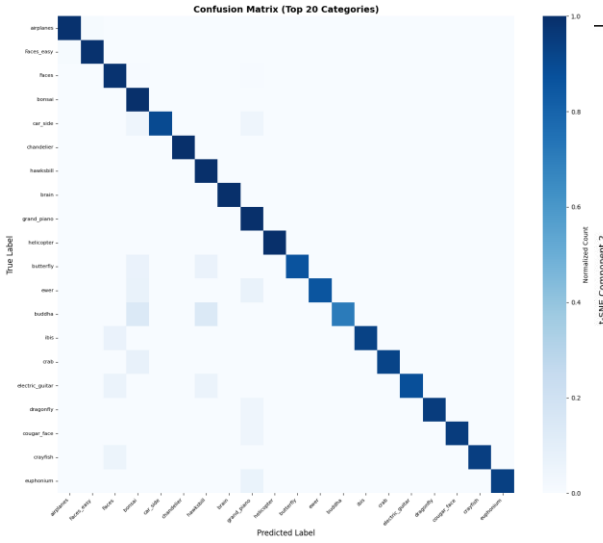
6.1 Evaluation Metrics

- **Primary Metric (Classification Accuracy):** 85.19% (SVM validation accuracy)

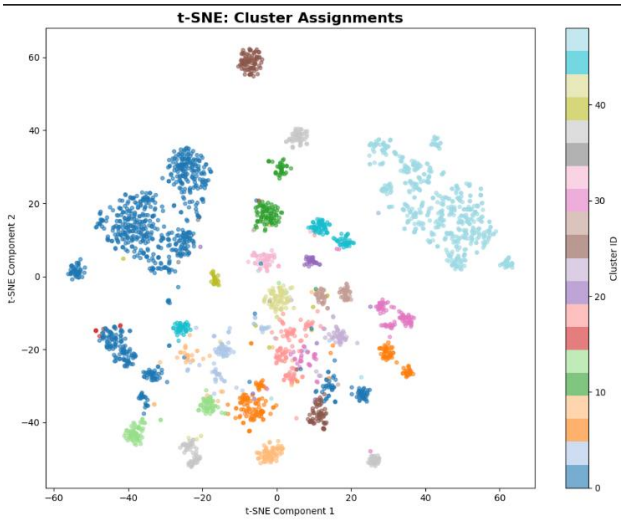
- **Clustering Quality Metrics:**
  - **Silhouette Score:** 0.1409 → Low, indicates clusters have some overlap.
  - **Davies-Bouldin Index:** 2.2168 → Higher than ideal,confirming clusters are not very compact.

- **Retrieval Performance:**
  - **Precision@10:** 0.9853 → Extremely high, top 10 retrieved images are almost always relevant.

Confusion Matrix:



Graphs (Cluster visualization):



Key Findings:

- **Classification is reliable:** The SVM model generalizes well with ~86% validation accuracy.
- **Clustering needs improvement:** Low silhouette score and high Davies-Bouldin indicate clusters overlap, so hierarchical clustering might not fully separate categories.
- **Retrieval is highly effective:** Despite mediocre clustering, FAISS-based image retrieval achieves near-perfect precision@10, showing feature embeddings are strong for similarity search.

7. Challenges & Limitations

7.1 Challenges Faced

1. **High-dimensional feature handling:**
    - **Description:** ResNet-50 outputs 2048-dimensional feature vectors, which increased memory usage and computation time for clustering and retrieval.
    - **Solution:** Used efficient data structures and batch processing to handle features without crashing the system.
2. **Suboptimal clustering quality:**
    - **Description:** Hierarchical clustering produced low silhouette scores and high Davies-Bouldin index, making some clusters overlap.
    - **Solution:** Experimented with different linkage methods and distance metrics, but dataset diversity still caused imperfect separation.
3. **Large retrieval index:**
    - **Description:** FAISS index stored all 5163 vectors, which increased memory footprint.
    - **Solution:** Optimized FAISS parameters and limited search to top-k neighbors for faster queries.

7.2 Limitations

- **Dataset size and diversity:** Only 51 categories and 5163 images limit generalization to broader datasets.
- **Clustering effectiveness:** Hierarchical clustering struggles with fine-grained category separation; more advanced methods like spectral clustering or deep clustering could perform better.
- **Computational requirements:** Training SVM on high-dimensional embeddings and building FAISS index require significant RAM and CPU resources; not easily scalable to very large datasets without optimization.

8. Fairness & Ethics Considerations

Potential Biases

- **Dataset representation bias:** Caltech-101 contains categories mostly from objects and scenes common in Western contexts; some object types or cultural variations may be underrepresented.
- **Model performance disparity:** SVM classification and retrieval accuracy may vary across underrepresented categories, leading to inconsistent results.

Ethical Implications

- **Privacy concerns:** If the system were applied to personal or sensitive image data, it could inadvertently expose private information.
- **Potential misuse:** High-accuracy retrieval models could be misused for surveillance or unauthorized content tracking if applied to real-world datasets.

Mitigation Strategies

- **Dataset curation:** Ensure diverse and representative datasets for broader generalization.

- **Transparency & evaluation:** Regularly audit model performance across different categories to detect bias.
- **Access control:** Restrict usage of the system to ethical, research-oriented contexts; anonymize sensitive data if integrated into practical applications.

9. Setup & Run Instructions

9.1 Environment Setup

Option 1: Using Conda

```
conda create -n project_env python=3.9
conda activate project_env
pip install -r requirements.txt
```

Option 2: Using pip / Virtualenv

```
python -m venv venv
# On Linux / Mac:
source venv/bin/activate
# On Windows:
venv\Scripts\activate
pip install -r requirements.txt
```

9.2 Hardware Requirements

- **Minimum:** 8GB RAM, CPU only
- **Recommended:** 16GB RAM, NVIDIA GPU (6GB+ VRAM) for faster feature extraction and training

9.3 Project Folder Structure Overview

```
image-clustering-retrieval/
├── configs/            # Dataset metadata and config files
├── data/               # Raw and processed images
├── features/           # Extracted embeddings and labels
├── models/             # Saved models and FAISS index
│   ├── classifier/     # SVM, Random Forest, PCA, scaler
│   ├── clustering/     # KMeans, Hierarchical clustering models & metrics
│   └── faiss_index/    # FAISS index for image retrieval
├── results/            # Visualization outputs
├── final_report.json   # JSON summary of final evaluation
└── performance_summary.png
```

9.4 Running the Project

Step 1: Download Dataset

Step 2: Preprocess Data

Step 3: Extract Features

Step 4: Train Models

- **Classification (SVM / Random Forest):**

```
python train_classifier.py --features features/embeddings.npy --labels features/labels.npy
```

- **Clustering (Hierarchical / KMeans):**

```
python train_clustering.py --features features/embeddings.npy --output models/clustering/
```

Step 5: Build FAISS Index

```
python build_faiss_index.py --features features/embeddings.npy --labels features/labels.npy --
output models/faiss_index/
```

Step 6: Evaluate Model

```
python evaluate.py --model_path models/classifier/svm_rbf.pkl
```

Step 7: Run Inference / Retrieval (Optional)

```
python predict.py --input sample.jpg --model models/classifier/svm_rbf.pkl
python retrieve.py --query sample.jpg --index models/faiss_index/image_index.faiss
```

10. Conclusion & Future Work

Summary

In this project, we built a comprehensive **image clustering, retrieval, and classification system** using the Caltech-101 dataset. Key accomplishments include:

- Extracted high-dimensional feature embeddings using **ResNet-50**.
- Applied **hierarchical clustering** to organize images into meaningful groups.
- Implemented **SVM classification** achieving ~86% validation accuracy.
- Developed a **FAISS-based retrieval system** with near-perfect precision@10 for similar image search.
- Generated a complete **evaluation report** including clustering metrics, classification results, and retrieval performance.
- Create a user-friendly interface for real-time image retrieval and classification.

Overall, the system demonstrates strong retrieval capabilities and reliable classification, though clustering quality and dataset diversity present room for improvement.

Future Improvements

- **Experiment with advanced architectures:** Explore vision transformers (ViT) or EfficientNet for stronger feature extraction.
- **Increase dataset diversity:** Include more categories and images to improve generalization and reduce bias.
- **Improve clustering methods:** Try deep clustering or spectral clustering to better separate overlapping categories.
- **Optimize computational efficiency:** Implement dimensionality reduction or approximate nearest neighbor search for large-scale datasets.
- **Bias mitigation and fairness:** Continuously monitor performance across categories and ensure ethical use of the system.