Mid Term Report: Image Retrieval

Harshita Vachhani (B23EE1026)
Prajna Agrawal (B23CS1054)
Sonam Sikarwar (B23CM1060)
Sreenitya Thatikunta (B23CS1072)
Nishkarsh Verma (B23CM1028)

March 31, 2025

Abstract

This project focuses on developing an Image Retrieval system to retrieve relevant images from a given query using the CIFAR-10 dataset. Features are extracted using Histogram of Oriented Gradients (HoG) and Convolutional Neural Network (CNN) architectures through the provided implementation. Various approaches, including Convolutional Neural Networks (CNN), Linear Discriminant Analysis (LDA), and Decision Trees, are employed to improve retrieval accuracy and efficiency. Initial trials involve evaluating the performance of these techniques independently and in combination.

Contents

1 Introduction

Image Retrieval is a fundamental computer vision task that involves retrieving images similar to a given query from a large database. It has various applications, including content-based image search, facial recognition, medical imaging, and object detection.

The CIFAR-10 dataset, used in this project, is a widely used benchmark dataset in machine learning and computer vision research. It consists of 60,000 color images divided into 10 distinct classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. Each class contains 6,000 images of size 32x32 pixels, with 50,000 images allocated for training and 10,000 for testing.

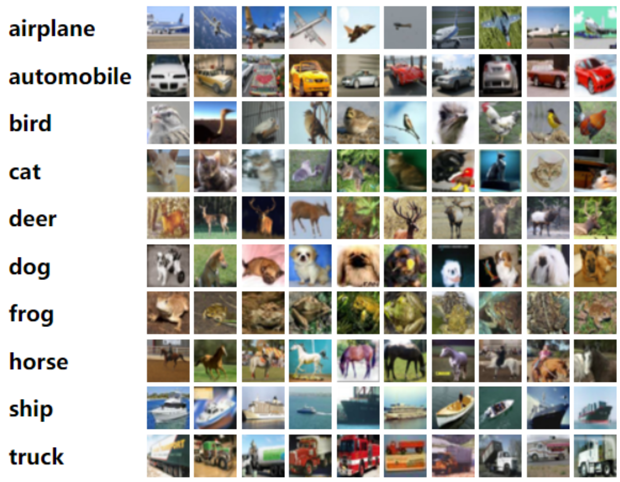

Figure 1: Example of CIFAR-10 images from various classes.

2 Approaches Explored

2.1 Decision Tree (DT)

A Decision Tree learns simple decision rules inferred from the data features to predict the label of a given image. As a baseline approach, a Decision Tree was trained directly on raw pixel values of CIFAR-10 images.

2.2 Decision Tree with HoG Features (DT + HoG)

In this approach, Histogram of Oriented Gradients (HoG) was used for feature extraction before training a Decision Tree. HoG captures edge and gradient information from images, providing a more descriptive representation compared to raw pixel values. The Decision Tree was trained on the extracted HoG features to evaluate its effectiveness for image retrieval.

2.3 Decision Tree with Convolutional Neural Network Features (CNN + DT)

A Convolutional Neural Network (CNN) was used to extract high-level features from images. The trained CNN's penultimate layer features were used as input to a Decision Tree for classification. This approach aimed to leverage the representational power of CNNs to enhance retrieval performance.

2.4 LDA with CNN and Decision Tree (LDA + CNN + DT)

Linear Discriminant Analysis (LDA) was applied to reduce the dimensionality of the CNN-extracted features before feeding them to a Decision Tree. LDA aims to find a linear combination of features that best separates the classes, potentially improving the performance of the Decision Tree by reducing noise and irrelevant information.

2.5 PCA with Decision Tree (PCA + DT)

Principal Component Analysis (PCA) was employed to reduce the dimensionality of raw image data before training a Decision Tree. This approach aimed to reduce computational complexity while retaining critical information for classification.

2.6 PCA with CNN and Decision Tree (PCA + CNN + DT)

Features extracted by a CNN were reduced using PCA before training a Decision Tree. PCA retained the most informative components, improving efficiency and potentially enhancing classification performance.

2.7 HoG with K-Means Clustering (HoG + KMeans)

Features extracted using HoG were clustered using the K-Means algorithm. The goal was to group similar images together based on their HoG features. K-Means clustering was explored as an unsupervised approach for image retrieval.

2.8 HoG with LDA and K-Means Clustering (HoG + LDA + KMeans)

To improve clustering performance, LDA was applied to HoG features before performing K-Means clustering. This dimensionality reduction technique aimed to enhance the separability of classes, thereby facilitating better clustering results.

2.9 K-Means with CNN (CNN + KMeans)

In this approach, the K-Means clustering algorithm is applied to features extracted using a Convolutional Neural Network (CNN). CNNs provide high-level feature representations, resulting in better clustering performance compared to HoG-based methods.

2.10 LDA with CNN and K-Means (LDA + CNN + KMeans)

This technique combines K-Means clustering with CNN feature extraction and LDA. The LDA step reduces the dimensionality of the CNN features, potentially improving clustering accuracy and reducing computational complexity.

2.11 HoG with Support Vector Machine (HoG + SVM)

A Support Vector Machine (SVM) was trained on HoG-extracted features to perform image classification. SVMs are effective classifiers that can handle high-dimensional data well, making them suitable for the HoG feature representation.

2.12 CNN with Support Vector Machine (CNN + SVM)

Features extracted by a CNN were classified using an SVM, combining CNN's feature extraction strength with SVM's classification efficiency.

2.13 K-Nearest Neighbors (KNN)

The KNN algorithm was applied for classification by comparing new data points to the most similar points in the training set. It's simple, effective, and works well for smaller datasets.

3 Results

3.1 Decision Tree Results

Test Accuracy: 0.3063

	precision	recall	f1-score	support
0	0.39	0.43	0.41	1000
1	0.35	0.29	0.32	1000
2	0.22	0.19	0.20	1000
3	0.17	0.18	0.18	1000
4	0.26	0.25	0.25	1000
5	0.28	0.22	0.24	1000
6	0.28	0.44	0.35	1000
7	0.31	0.28	0.29	1000
8	0.44	0.44	0.44	1000
9	0.36	0.35	0.35	1000
accuracy			0.31	10000
macro avg	0.31	0.31	0.30	10000
weighted avg	0.31	0.31	0.30	10000

Figure 2: DT

Test Accuracy: 0.2487

	precision	recall	f1-score	support
0	0.26	0.19	0.22	1000
1	0.38	0.31	0.34	1000
2	0.17	0.21	0.19	1000
3	0.17	0.11	0.13	1000
4	0.19	0.30	0.23	1000
5	0.21	0.20	0.20	1000
6	0.28	0.31	0.29	1000
7	0.29	0.22	0.25	1000
8	0.31	0.36	0.34	1000
9	0.26	0.28	0.27	1000
accuracy			0.25	10000
macro avg	0.25	0.25	0.25	10000
weighted avg	0.25	0.25	0.25	10000

Figure 3: DT + HoG

Decision Tree Test Accuracy on CNN features: 0.6719

	precision	recall	f1-score	support
0	0.68	0.67	0.68	1000
1	0.79	0.78	0.78	1000
2	0.59	0.56	0.57	1000
3	0.48	0.62	0.54	1000
4	0.57	0.61	0.59	1000
5	0.69	0.63	0.66	1000
6	0.74	0.68	0.71	1000
7	0.68	0.59	0.63	1000
8	0.79	0.79	0.79	1000
9	0.78	0.79	0.79	1000
accuracy			0.67	10000
macro avg	0.68	0.67	0.67	10000
weighted avg	0.68	0.67	0.67	10000

Figure 4: DT + CNN

Decision Tree Test Accuracy after LDA: 0.8755

	precision	recall	f1-score	support
0	0.89	0.89	0.89	1000
1	0.94	0.93	0.94	1000
2	0.86	0.82	0.84	1000
3	0.78	0.77	0.77	1000
4	0.82	0.88	0.85	1000
5	0.85	0.83	0.84	1000
6	0.90	0.89	0.89	1000
7	0.87	0.88	0.87	1000
8	0.94	0.94	0.94	1000
9	0.93	0.92	0.92	1000
accuracy			0.88	10000
macro avg	0.88	0.88	0.88	10000
weighted avg	0.88	0.88	0.88	10000

Figure 5: LDA + CNN + DT

Test Accuracy after PCA: 0.3112

	precision	recall	f1-score	support
0	0.39	0.46	0.42	1000
1	0.37	0.33	0.35	1000
2	0.25	0.27	0.26	1000
3	0.19	0.20	0.20	1000
4	0.25	0.29	0.27	1000
5	0.28	0.25	0.26	1000
6	0.33	0.34	0.33	1000
7	0.33	0.22	0.27	1000
8	0.42	0.40	0.41	1000
9	0.32	0.34	0.33	1000
accuracy			0.31	10000
macro avg	0.31	0.31	0.31	10000
weighted avg	0.31	0.31	0.31	10000

Figure 6: PCA + DT

Test Accuracy after PCA: 0.7047

	precision	recall	f1-score	support
0	0.70	0.74	0.72	1000
1	0.81	0.81	0.81	1000
2	0.64	0.56	0.59	1000
3	0.55	0.61	0.57	1000
4	0.63	0.63	0.63	1000
5	0.65	0.69	0.67	1000
6	0.75	0.75	0.75	1000
7	0.72	0.65	0.68	1000
8	0.80	0.80	0.80	1000
9	0.83	0.81	0.82	1000
accuracy			0.70	10000
macro avg	0.71	0.70	0.70	10000
weighted avg	0.71	0.70	0.70	10000

Figure 7: PCA + CNN + DT

3.2 K-Means Results

	precision	recall	f1-score	support
0	0.00	0.00	0.00	1000
1	0.28	0.51	0.36	1000
2	0.24	0.19	0.21	1000
3	0.00	0.00	0.00	1000
4	0.00	0.00	0.00	1000
5	0.30	0.40	0.34	1000
6	0.20	0.43	0.27	1000
7	0.19	0.39	0.26	1000
8	0.36	0.30	0.33	1000
9	0.32	0.33	0.33	1000
accuracy			0.26	10000
macro avg	0.19	0.26	0.21	10000
weighted avg	0.19	0.26	0.21	10000

Figure 8: K-Means + HoG

	precision	recall	f1-score	support
0	0.66	0.49	0.56	1000
1	0.62	0.65	0.63	1000
2	0.50	0.32	0.39	1000
3	0.41	0.31	0.35	1000
4	0.38	0.58	0.46	1000
5	0.42	0.40	0.41	1000
6	0.45	0.60	0.52	1000
7	0.54	0.57	0.56	1000
8	0.60	0.55	0.58	1000
9	0.61	0.64	0.63	1000
accuracy			0.51	10000
macro avg	0.52	0.51	0.51	10000
weighted avg	0.52	0.51	0.51	10000

Figure 9: K-Means + HoG + LDA

	precision	recall	f1-score	support
0	0.82	0.61	0.70	1000
1	0.83	0.84	0.83	1000
2	0.78	0.46	0.58	1000
3	0.45	0.61	0.52	1000
4	0.60	0.58	0.59	1000
5	0.63	0.69	0.66	1000
6	0.68	0.78	0.72	1000
7	0.77	0.63	0.69	1000
8	0.71	0.86	0.78	1000
9	0.80	0.84	0.82	1000
accuracy			0.69	10000
macro avg	0.71	0.69	0.69	10000
weighted avg	0.71	0.69	0.69	10000

Figure 10: K-Means + CNN

	precision	recall	f1-score	support
0	0.89	0.94	0.91	1000
1	0.96	0.92	0.94	1000
2	0.92	0.82	0.87	1000
3	0.70	0.89	0.78	1000
4	0.83	0.90	0.86	1000
5	0.92	0.78	0.85	1000
6	0.92	0.90	0.91	1000
7	0.95	0.87	0.91	1000
8	0.95	0.95	0.95	1000
9	0.94	0.93	0.93	1000
accuracy			0.89	10000
macro avg	0.90	0.89	0.89	10000
weighted avg	0.90	0.89	0.89	10000

Figure 11: K-Means + CNN + LDA

3.3 SVM Results

	precision	recall	f1-score	support
0	0.57	0.62	0.59	1000
1	0.59	0.64	0.61	1000
2	0.45	0.41	0.43	1000
3	0.41	0.33	0.37	1000
4	0.43	0.48	0.45	1000
5	0.46	0.42	0.44	1000
6	0.53	0.62	0.57	1000
7	0.59	0.56	0.58	1000
8	0.58	0.58	0.58	1000
9	0.65	0.64	0.65	1000
accuracy			0.53	10000
macro avg	0.53	0.53	0.53	10000
weighted avg	0.53	0.53	0.53	10000

Figure 12: SVM + HoG

	precision	recall	f1-score	support
0	0.86	0.88	0.87	1000
1	0.93	0.92	0.92	1000
2	0.81	0.83	0.82	1000
3	0.74	0.77	0.75	1000
4	0.83	0.86	0.84	1000
5	0.84	0.80	0.82	1000
6	0.89	0.88	0.88	1000
7	0.91	0.86	0.88	1000
8	0.91	0.92	0.92	1000
9	0.92	0.91	0.91	1000
accuracy			0.86	10000
macro avg	0.86	0.86	0.86	10000
weighted avg	0.86	0.86	0.86	10000

Figure 13: SVM + CNN

3.4 KNN Results

	precision	recall	f1-score	support
0	0.38	0.54	0.45	1000
1	0.67	0.21	0.32	1000
2	0.22	0.44	0.29	1000
3	0.30	0.23	0.26	1000
4	0.25	0.52	0.33	1000
5	0.39	0.22	0.28	1000
6	0.36	0.26	0.30	1000
7	0.69	0.22	0.33	1000
8	0.40	0.66	0.50	1000
9	0.73	0.13	0.23	1000
accuracy			0.34	10000
macro avg	0.44	0.34	0.33	10000
weighted avg	0.44	0.34	0.33	10000

Figure 14: KNN

4 Future Directions

Moving forward, we plan to explore a variety of techniques to enhance our project's performance. We aim to implement perceptrons and artificial neural networks (ANNs) to capture more complex patterns in the data. Additionally, we will experiment with Gaussian Mixture Models (GMM) for effective data clustering. To improve prediction accuracy, we intend to apply linear regression for continuous outputs and logistic regression for classification tasks. Incorporating random forests could provide more robust results by aggregating multiple decision trees. Furthermore, we will explore Bayesian methods to improve predictions by learning from prior knowledge. By testing these approaches and potentially combining them, we hope to achieve better overall performance.

5 Contributions

1. Harshita Vachhani - Decision Tree Models
2. Sonam Sikarwar - K-Means Models
3. Prajna Agrawal - SVM Models
4. Sreenitya Thatikunta - Report + Decision Tree Models
5. Nishkarsh Verma - KNN Model