Image Retrieval

PRML CSL2050 Course Project

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

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), Principal Component Analysis (PCA), K Nearest Neighbour (KNN), Logistic Regression, Random Forests, Kmeans Clustering are employed to improve retrieval accuracy and efficiency.

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.

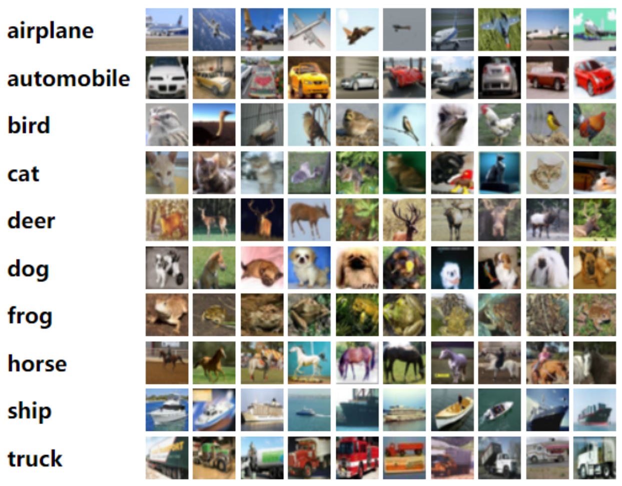

Figure 1: Example of CIFAR-10 images from various classes.

2 Experiments

Approach Name	Accuracy (%)
Decision Tree	31
Decision Tree + HoG features	25
Decision Tree + CNN features	67
Decision Tree + LDA + CNN features	88
Decision Tree + PCA	31
Decision Tree + PCA + CNN features	70
K-Means + HoG features	26
K-Means + LDA + HoG features	51
K-Means + CNN features	69
K-Means + LDA + CNN features	89
SVM + HoG features	53
SVM + CNN features	86
KNN	34

Table 1: Comparison of Approaches by Accuracy

3 Models

3.1 ResNet-32 Model

ResNet-32 is a convolutional neural network architecture that leverages residual learning to facilitate the training of deep networks. It is especially well-suited for image classification tasks like CIFAR-10.

The ResNet-32 model consists of the following components:

- **Initial Convolution:** A single 3×3 convolutional layer with 16 output channels, followed by batch normalization and ReLU activation.
- **Residual Blocks:** The network has three stages of residual blocks:
  - **Stage 1:** 5 residual blocks with 16 output channels each.
  - **Stage 2:** 5 residual blocks with 32 output channels. The first block in this stage downsamples the input using a stride of 2.
  - **Stage 3:** 5 residual blocks with 64 output channels. Similar to Stage 2, the first block downsamples the input.

Each residual block contains two 3×3 convolutional layers, with batch normalization and ReLU activation applied after each convolution. If the input and output dimensions differ, a projection shortcut using a 3×3 convolution and batch normalization is applied to the residual.

- **Global Average Pooling:** An 8×8 average pooling layer is used to reduce the spatial dimensions of the feature maps.
- **Fully Connected Layer:** The output of the pooling layer is flattened and passed through a linear layer to produce the final classification logits for 10 CIFAR-10 classes.

Feature Extraction and Dimensionality Reduction

The following three models — Aether, Nuvora, and Orion — share a common preprocessing pipeline comprising deep feature extraction and dimensionality reduction.

Deep Feature Extraction with ResNet50

To capture rich visual representations, raw CIFAR-10 images are passed through a pre-trained ResNet50 model. This deep convolutional neural network, originally trained on the ImageNet dataset, acts as a powerful feature extractor. The final classification layer is removed, and the output of the penultimate layer is used as a compact, high-level feature vector for each image. These vectors encapsulate spatial and semantic information crucial for downstream learning tasks.

Dimensionality Reduction with Linear Discriminant Analysis (LDA)

The extracted feature vectors are then projected into a lower-dimensional space using Linear Discriminant Analysis (LDA). LDA is a supervised dimensionality reduction technique that maximizes class separability by projecting data along axes that best discriminate between classes. Given the 10-class CIFAR-10 dataset, LDA reduces the feature space to 9 dimensions (i.e., `num_classes - 1`), improving computational efficiency while preserving class-specific variance. These reduced representations are then used by the subsequent classifiers for training and inference.

3.2 Aether: Logistic Regression

Aether is a supervised classification model that utilizes multinomial logistic regression to classify data into one of several possible classes. It is particularly well-suited for multi-class problems where the goal is to model the probability of each class given a set of input features.

Methodology

- **Model Formulation:** Aether employs multinomial logistic regression, which extends binary logistic regression to handle multiple classes. It estimates the probability of a sample belonging to each class using the softmax function, which ensures the output probabilities sum to 1.
- **Optimization:** The model is trained using the `lbfgs` solver, a quasi-Newton optimization algorithm well-suited for convex problems. A maximum of 1000 iterations is allowed to ensure convergence during training.
- **Training:** The model learns a separate weight vector for each class by minimizing the cross-entropy loss between predicted and actual class probabilities.
- **Prediction:** During inference, the model computes the probability of each class and assigns the sample to the one with the highest probability.

3.3 Nuvora: Random Forest Classifier

Nuvora leverages the power of ensemble learning through a Random Forest Classifier to perform multi-class classification. Random Forest is a collection of decision trees trained on different parts of the data and aggregated to improve generalization.

Methodology

- **Model Structure:** Nuvora constructs an ensemble of 100 decision trees, each trained on a bootstrapped subset of the data. The diversity among trees ensures robustness and reduces overfitting.
- **Tree Constraints:**
  - Maximum tree depth is set to 15, limiting model complexity.
  - Each internal node must have at least 5 samples to consider a split (`min_samples_split=5`).
  - Leaf nodes must contain at least 3 samples (`min_samples_leaf=3`), improving model regularization.
- **Prediction:** For inference, each tree predicts a class label, and the final prediction is made via majority voting across all trees.

3.4 Orion: K-Means Clustering

Orion is an unsupervised learning model that applies the K-Means clustering algorithm to group data into distinct categories. Unlike supervised models that rely on labeled data, Orion identifies inherent patterns by organizing inputs into k clusters, where k corresponds to the number of desired output categories.

Methodology

- **Initialization:** Orion begins by randomly selecting k initial cluster centroids. These centroids represent the center of each cluster and are refined throughout the training process.
- **Assignment Step:** Each data point is assigned to the cluster with the nearest centroid based on Euclidean distance. This step groups similar data points together.
- **Update Step:** The centroids are recalculated as the mean of all data points assigned to that cluster. This process is repeated iteratively until the centroids converge or a maximum number of iterations is reached.
- **Cluster Labeling:** After clustering, each cluster is mapped to a class label by analyzing the most common true label among its members (using majority voting). This step enables evaluation against labeled test data.
- **Prediction:** During inference, new data points are assigned to the nearest cluster, and the corresponding mapped label is used as the predicted class.

4 Retrieval Pipeline

The system includes a retrieval mechanism that allows for identifying visually similar examples based on class predictions. The process is as follows:

- A **query image** is uploaded by the user.
- The **selected classification model** (Vortex, Aether, Nuvora, or Orion) is used to predict the class of the query image. This becomes the *target class*.
- The **test dataset** is treated as the retrieval database.
- The system iterates over the test images and identifies the first five images whose predicted class matches the target class.
- These five images are returned as the output, representing examples visually and semantically similar to the query image.

This method provides intuitive visual feedback for the classification decision and serves as a simple yet effective content-based image retrieval (CBIR) mechanism.

5 Results

The classification accuracy achieved by each model on the CIFAR-10 test set is summarized below:

Model	Accuracy (%)
Vortex (ResNet32)	89
Aether (Logistic Regression)	91
Orion (K-Means Clustering)	91
Nuvora (Random Forest)	91

Table 2: Test accuracy of different models on CIFAR-10

6 Deployment

The deployment of the models was carried out using cloud-based resources, specifically Google Cloud's Computing Engine, to ensure scalable and efficient execution of the models in a production environment.

- **Google Cloud Computing Engine:** The models were deployed on a virtual machine created within Google Cloud's Compute Engine. Google Cloud provides an easy-to-use platform for managing virtual machines (VMs) that can scale up and down according to resource requirements, making it ideal for model deployment in an agile environment. The virtual machine provides a reliable infrastructure for hosting and serving the models in a high-performance environment with guaranteed uptime.

- **Streamlit Deployment:** Streamlit was used to create an interactive web interface for the model deployment. With Streamlit, we were able to develop a user-friendly web application that allowed users to easily interact with the models. The interface enables users to upload images, select models for inference, and retrieve results with minimal configuration. The integration with Google Cloud ensures that the models are accessible remotely, enabling users to interact with them through the web-based interface.

This cloud-based deployment approach ensured that the models could be efficiently run at scale, accessed remotely for real-time predictions, and maintained with ease, all while taking advantage of Google Cloud's infrastructure for computationally intensive tasks.

7 Contributions

- Harshita Vachhani - Nuvora + Deployment
- Sonam Sikarwar - Orion + Spotlight Video
- Prajna Agrawal - Vortex + Deployment
- Sreenitya Thatikunta - Aether + Documentation (Report)
- Nishkarsh Verma - Experiment (KNN Models)

8 Bibliography

- Medium
- Scikit-learn Documentation
- Streamlit Documentation
- PyTorch Documentation