CSL2050 - Pattern Recognition and Machine Learning

Object Recognition - CIFAR10

Team Member

Rohan Regar
Arnava Srivastava
Buddhav Garg
Dheeraj Dhakar
Harshit Barwar

Anand Mishra
Assistant Professor, Dept of CSE
IIT Jodhpur

Abstract

The project aimed to optimize and enhance the performance of machine learning classifiers on the CIFAR-10 dataset, a well-known benchmark dataset for image classification. Various techniques were explored, including feature extraction using ResNet-50, Histogram of Oriented Gradients (HoG), and hyperparameter tuning for classification algorithms such as K-Nearest Neighbors (KNN), Decision Trees, and Naive Bayes.

The findings indicate that leveraging deep learning features, especially from models like ResNet-50, coupled with hyperparameter optimization, can substantially enhance the image classification performance on the CIFAR-10 dataset. The study underscores the significance of feature engineering and model tuning in achieving better results for image classification tasks.

Keywords: Image Classification, CIFAR-10 Dataset, Machine Learning, ResNet-50, Hyperparameter Tuning.

Contents

1 Introduction

Problem Statement:
In this project, we aimed to explore various machine learning techniques for image classification on the CIFAR-10 dataset. The CIFAR-10 dataset consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class. Our objective was to build and evaluate different machine learning models to classify these images into their respective categories.

Major Findings:

- Initial attempts with classical machine learning techniques like k-NN, decision trees, random forests, and Naive Bayes yielded moderate accuracy.
- Feature extraction using pretrained ResNet-50 CNN significantly improved classification accuracy, achieving up to 80
- Parameter optimization and failure case analysis further refined the models, providing insights into misclassifications.

Structure of Code File:

Data Preprocessing:

- Loading and visualizing the CIFAR-10 dataset.
- Data normalization and flattening.

Testing Classical Machine Learning Models:

- KNN, Decision Trees, Random Forest, and Naive Bayes classifiers were trained and evaluated on the CIFAR-10 dataset.

Techniques to Improve Accuracy:

- Linear Discriminant Analysis (LDA): Used to maximize class separation.
- Principal Component Analysis (PCA): Employed for dimensionality reduction.
- Feature Extraction Using ResNet-50.
- Histogram of Oriented Gradients (HOG): Feature extraction method to capture image gradients.

Performance Comparison with Feature Extraction:

- Classical machine learning models were retrained using the features extracted by LDA, PCA, and HOG techniques.
- Failure case analysis using misclassification.

1.1 References

- Utilized feature extraction code from professors Github Repo.`https://github.com/anandmishra22/PRML-Spring-2023/blob/main/Project/Reference_Codes/refCodes4PRMLProject.ipynb`
- Learned the basics to implement HoG from this page. `https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/`
- Understanding of ResNet-50 from following Link. `https://medium.com/@sharma.tanish096/detailed-explanation-of-residual-network-resnet50-cnn-model-106e0ab9fa9e`
- Hyperparameter Tuning from following link are used to understand while Implementaion. `https://www.anyscale.com/blog/what-is-hyperparameter-tuning`

2 Approach

Classical ML Techniques: We explored several classical machine learning techniques:

- k-NN (k-Nearest Neighbors): We experimented with k-NN while optimizing parameters to find the optimal number of neighbors for classification.
- Decision Tree: Decision tree-based methods were used for classification tasks.
- Random Forest: Ensemble techniques like random forests were employed to improve classification accuracy by combining multiple decision trees.
- Naive Bayes: We implemented Naive Bayes, a probabilistic classifier commonly used in machine learning.
- SVM (Support Vector Machine): Model used for classification and regression tasks, aiming to find the optimal hyperplane that best separates data points into different classes

Dimensionality Reduction: We applied Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) to reduce the dimensionality of the feature space. However, no significant improvements were observed with these techniques.

Feature Extraction using Pretrained CNN (ResNet-50): We utilized a pretrained Convolutional Neural Network (CNN) architecture, ResNet-50, to extract features from the images. Features were extracted from the last layer of the ResNet-50 model, capturing high-level image representations.

Histogram of Oriented Gradients (HoG): We employed the Histogram of Oriented Gradients (HoG) technique as a feature descriptor to capture the local shape and edge information within the images. HoG features were computed to represent the distribution of gradient orientations in different image regions, providing valuable information for classification tasks.

3 Experiments and Results

3.1 CIFAR-10 Dataset

1. Overview The CIFAR-10 dataset consists of 60,000 32x32 color images in 10 classes, with 6,000 images per class.
2. Data Format The dataset is provided in a binary format, where each image is a 32x32 RGB image, resulting in a shape of (60000, 32, 32, 3).
3. Reference The dataset and methodology followed are described in greater detail in the following paper: `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.
4. Loading Data The dataset can be loaded using the unpickle method to access the training and test sets separately.

3.2 Data Preprocessing

5. Normalization Normalization is performed using min-max scaling to scale the pixel values from the range 0-255 to 0.0-1.0 (feature scaling). This is done because the pixel values in the RGB channels of the feature set range from 0-255.
6. Flattening Flattening is typically done as traditional machine learning models require input in 2-D/tabular format.

3.3 Preliminary Tests

We apply KNN, Decision Tree, Random Forest, Naive Bayes, SVM, ANN and following are the results of these Traditional Models

Table 1: Performance metrics measures for classical techniques -

MODELS	Accuracy	Precision	Recall	F1-score
KNN	0.339	0.430	0.339	0.326
Decision Tree	0.2673	0.267	0.2673	0.267
Random Forest	0.472	0.468	0.472	0.470
Naive Bayes	0.297	0.311	0.297	0.275
SVM	0.544	0.542	0.544	0.542
ANN	0.476	0.489	0.476	0.472

Resons for the failure of different Classical Machine Learning Models :

1. K-NN

- Curse of Dimensionality: CIFAR-10 images have a high dimensionality (32x32x3 = 3072 features). KNN struggles with high-dimensional data due to the curse of dimensionality.
- Pixel Values: RGB pixel values between 0 and 255 are treated as independent features. This approach doesn't capture spatial hierarchies and relationships between pixels.

2. Decision Tree

- Overfitting: Decision trees can easily overfit to training data, especially with high-dimensional data like images.
- Lack of Feature Relationships: Decision trees make decisions based on individual features without considering the relationships between them, making it hard to capture complex patterns in images.

3. Random Forest Classifier

- Complexity: Even though random forests can handle high-dimensional data better than decision trees, they might still struggle with capturing spatial hierarchies and intricate relationships in images.
- Ensemble of Weak Learners: Random forests are ensembles of decision trees, which might not be ideal for capturing the complexities of image data without deep feature extraction.

4. Naive Bayes Classifier

- Independence Assumption: Naive Bayes assumes that features are independent, which is not the case for image data. Pixels in images are spatially correlated, and their relationships are crucial for classification.
- Continuous Features: Naive Bayes works well with categorical features but struggles with continuous features like pixel values in images.

5. Support Vector Machine (SVM)

- High-Dimensional Data: SVMs perform well in low to medium-dimensional spaces but can become computationally expensive and inefficient in high-dimensional spaces like the CIFAR-10 dataset (32x32x3 = 3072 features).

- Linear Separability: SVM aims to find the hyperplane that best separates the data into different classes. In the case of CIFAR-10 images, the data might not be linearly separable in the feature space defined by raw pixel values. This can lead to suboptimal decision boundaries and reduced classification accuracy.
- Kernel Selection: While SVM can utilize various kernels to handle non-linear data, selecting an appropriate kernel and tuning its parameters can be challenging and computationally expensive for high-dimensional image data.

3.4 Methods Used to Improve Performance

Dimensionality Reduction: Techniques like PCA and LDA can be applied to reduce the dimensionality of the data, making it easier for classical ML models to handle.
Deep Learning and CNNs: Deep learning models like Convolutional Neural Networks (CNNs) are well-suited for image classification tasks. They can automatically learn hierarchical features from images, capturing spatial relationships and patterns.
Feature Engineering: Instead of using raw pixel values, feature engineering techniques like histogram of oriented gradients (HoG), or using pretrained models for feature extraction can be beneficial.
Hyperparameter Tuning: Optimize the hyperparameters of the classical ML models to find the best configuration for the CIFAR-10 dataset.

3.4.1 Performance After Dimensionality Reduction

After applying dimensionality reduction techniques like LDA (Linear Discriminant Analysis) and PCA (Principal Component Analysis), the performance of the machine learning models exhibited varied outcomes. Notably, there was a slight improvement in accuracy for the Naive Bayes classifier. However, the accuracy decreased for the Random Forest and SVM classifiers. The performance of the Decision Tree and K-Nearest Neighbors (KNN) classifiers remained relatively unchanged.

Table 2: Performance metrics measures after performing PCA -

MODELS	Accuracy	Precision	Recall	F1-Score
KNN	0.362	0.446	0.362	0.354
Decision Tree	0.257	0.257	0.257	0.257
Random Forest	0.434	0.429	0.434	0.430
Naive Bayes	0.312	0.333	0.312	0.299
SVM	0.470	0.506	0.470	0.478
ANN	0.520	0.520	0.520	0.514

Table 3: Performance metrics measures after performing LDA -

MODELS	Accuracy	Precision	Recall
KNN	0.329	0.334	0.329
Decision Tree	0.278	0.278	0.278
Random Forest	0.369	0.368	0.369
Naive Bayes	0.365	0.366	0.365
SVM	0.373	0.373	0.373

- **Improvement:** LDA and PCA might have enhanced the feature separability, making the data more discriminative, which could have benefited the Naive Bayes classifier that assumes feature independence.
- **Decrease in Accuracy:**
  - Loss of Information: Dimensionality reduction techniques like LDA and PCA might have resulted in the loss of some discriminative information crucial for the Random Forest and SVM classifiers, affecting their performance negatively.
  - Complexity Reduction: Random Forest and SVM, being inherently capable of handling high-dimensional data, might not benefit significantly from dimensionality reduction and could even suffer due to reduced feature space.

3.4.2 Performance After Using HoG

HoG is a feature descriptor that captures the distribution of gradients (edge orientations) in an image. It is used primarily in computer vision tasks for object detection and image classification.

Key Parameters:

- Orientations: Number of gradient orientations to consider (e.g., 9 orientations).
- Pixels per Cell: Size of the cell in pixels (e.g., 8x8 pixels).
- Cells per Block: Number of cells in each block (e.g., 2x2 cells).

We have used the template code provided for applying HoG.

Table 4: Performance metrics measures after HoG -

MODELS	Accuracy	Precision	Recall	F1-score
KNN	0.523	0.572	0.524	0.518
Decision Tree	0.279	0.281	0.279	0.280
Random Forest	0.521	0.518	0.521	0.516
Naive Bayes	0.458	0.466	0.458	0.454
SVM	0.632	0.632	0.632	0.631
ANN	0.479	0.479	0.470	0.471

- Improved Accuracy: The classifiers trained on HoG features exhibited improved accuracy compared to models trained on raw pixel values, albeit not as high as those trained on ResNet features.
- Enhanced Feature Representation: HoG captures texture and shape information, which might be beneficial for distinguishing between different CIFAR-10 classes compared to raw pixel values.
- It may not capture the intricate details and relationships present in the data as effectively as a deep neural network like ResNet.

3.4.3 Performance After Using ResNet-50

- ResNet, which stands for Residual Network, is a type of deep neural network architecture that was introduced to address the problem of vanishing gradients in very deep neural networks.
- It introduced the concept of residual learning to CNN architectures, and has a deep architecture making it relevant for tasks like image classification, object detection, and image segmentation.

- Transfer learning - We have used the template code provided for feature extraction from the last layer of a pre-trained ResNet-50 model.

After extracting features from the pre-trained ResNet-50 model and applying classical machine learning techniques, a significant improvement in accuracy was observed across multiple classifiers.

Table 5: Performance metrics measures after ResNet-50 -

MODELS	Accuracy	Precision	Recall	F1-score
KNN	0.841	0.844	0.841	0.840
Decision Tree	0.646	0.647	0.646	0.646
Random Forest	0.811	0.815	0.811	0.811
Naive Bayes	0.792	0.794	0.792	0.791
SVM	0.898			
Naive Bayes	0.878	0.878	0.878	0.880

Improved accuracy: The reason is simple - the features extracted from ResNet-50 are rich and high-dimensional, capturing intricate patterns and representations from the images. This solves the primary problem of capturing spatial correlations of the pixels, faced by the classical models.

3.4.4 Hyperparameter Tuning

While applying a pre-trained model like ResNet-50 is a powerful approach for the Cifar-10 dataset, there's still room for improvement through hyperparameter tuning on the final classification layer. Hyperparameter tuning is a crucial step in optimizing the performance of machine learning models. It involves selecting the best hyperparameters that govern the learning process of the model to achieve improved accuracy and generalization on unseen data.

K-Nearest Neighbors (KNN)

- By varying the number of neighbors (k), the accuracy of the KNN classifier was evaluated.
- The optimal value of k was determined using a range of k values from 3 to 19.
- The accuracy plot displayed the performance of the KNN classifier for each k value, with a peak accuracy achieved at the best k value.
- **Best k value = 13**

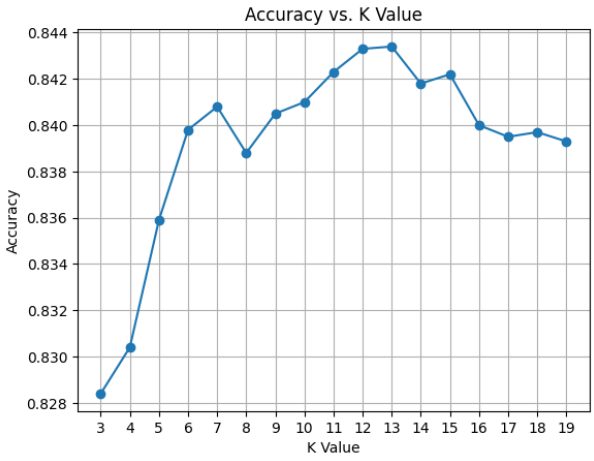


Figure 1: Accuracy Plot for different values of k

Decision Tree

- Two different criteria, "gini" and "entropy", were explored to split the nodes of the Decision Tree.
- Further hyperparameter tuning was conducted by setting constraints on the maximum depth of the tree, minimum samples leaf, and minimum samples split.
- Best parameter found from the grid
  - Max_depth : [8,10 , 12],
  - Min_samples_leaf : [8 ,10 ,12]
  - Min_samples_split : [13,15,17]
- **The Decision Tree classifier achieved the highest accuracy when the criterion was set to "entropy" with a maximum depth of 10, minimum samples leaf of 10, and minimum samples split of 15.**

Naive Bayes

- The **Multinomial** Naive Bayes classifier was optimized by tuning the alpha parameter, which represents the smoothing parameter.
- Grid Search Cross Validation was employed to find the optimal alpha value.
- The best alpha value obtained was 2.0, which resulted in an accuracy of 0.7915.
- The **Gaussian** Naive Bayes classifier was optimized by tuning the var_smoothing parameter, which adds a fraction of the largest variance of all features to variances for calculation stability.
- The best hyperparameters found for Gaussian Naive Bayes were 'var_smoothing': 1e-09 and accuracy achieved was 0.7948.

Support Vector Machine (SVM)

- The SVM classifier was fine-tuned by exploring different combinations of hyperparameters such as C, gamma, and kernel.
- Grid Search Cross Validation was utilized to identify the optimal hyperparameters.
- The best hyperparameters found were 'C': 2, 'gamma': 'scale', 'kernel': 'rbf', with an accuracy score of 0.891 and the best score of 0.896.

Table 6: **Results of Accuracy After Hyperparameter Tuning**

MODELS	Accuracy
KNN	0.843
Decision Tree	0.663
Naive Bayes	0.795
SVM	0.891

3.5 Misclassification Results

After evaluating the performance of various machine learning models, it's crucial to understand where these models are failing. Misclassification analysis provides insights into the specific classes that are often confused with each other by the models.

Figure 2: k-NN

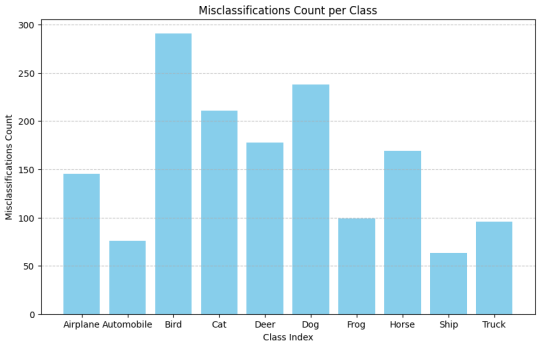


Figure 3: Decision Tree

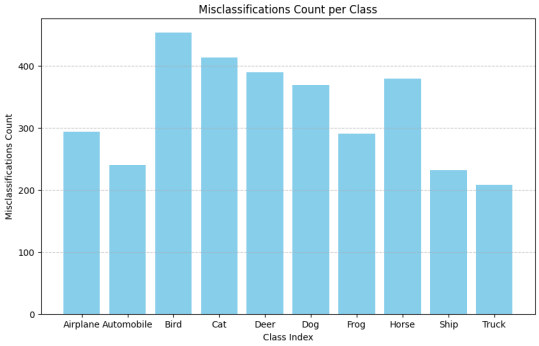


Figure 4: Random Forest

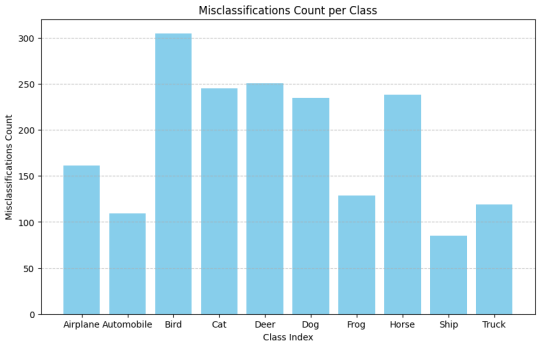


Figure 5: Naive Bayes

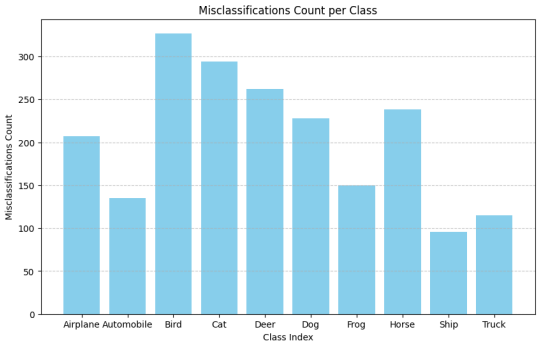


Figure 6: SVM

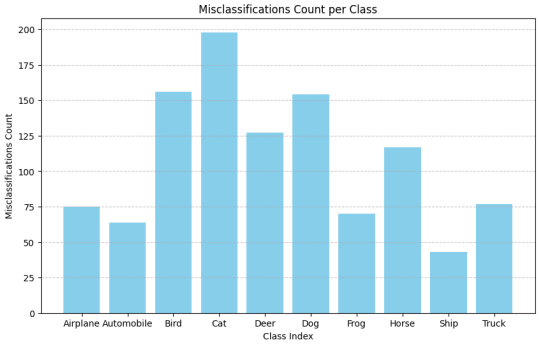
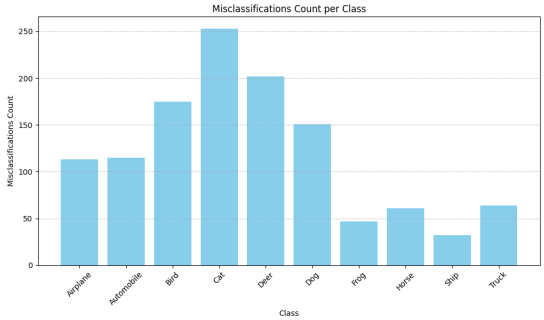


Figure 7: ANN



Inferences:

- Uniform Misclassifications: Models like Decision Tree and Random Forest seem to generalize well across various classes but might benefit from more nuanced feature engineering or deeper architectures to capture intricate class differences.
- Specific Confusions: Models like Naive Bayes and SVM show specific pairs of classes where confusion is more frequent. This indicates the need for targeted improvements in feature extraction or model architecture to resolve these issues.
- Balanced Misclassifications: KNN and ANN shows a balanced misclassification pattern, suggesting a more holistic understanding of the data.

4 Summary

The report investigates image classification on the CIFAR-10 dataset using classical machine learning techniques and advanced methods like feature extraction with pretrained CNNs. Initially, traditional models like k-NN, decision trees, and random forests yielded moderate accuracy. However, feature extraction with ResNet-50 significantly improved classification accuracy to 0.80. Attempts to improve classical models through dimensionality reduction and feature engineering had mixed results, with slight improvements for some models but decreased accuracy for others. Histogram of Oriented Gradients (HoG) showed promise, albeit not as effective as deep learning methods. Finally, hyperparameter tuning further improved accuracy across various classifiers. Misclassification analysis highlighted areas for improvement, suggesting the need for more nuanced feature engineering and model architecture adjustments. Overall, leveraging deep learning techniques like ResNet-50 for feature extraction proved most effective in enhancing classification accuracy on the CIFAR-10 dataset.

Future Prospects
As we delve deeper into the intricacies of the CIFAR-10 dataset, there are several avenues to explore for improving model performance:

- Advanced Feature Engineering: Techniques like HoG and deep feature extraction (e.g., from pretrained models like ResNet) have shown promise. Exploring more advanced feature engineering methods tailored to image data could yield significant improvements.
- Ensemble Methods: Combining the strengths of multiple models through ensemble methods can often lead to better generalization and reduced overfitting. Techniques like stacking or boosting could be explored.
- Neural Network Architectures: Experimenting with more complex neural network architectures, including convolutional neural networks (CNNs), can help in capturing intricate patterns in image data, potentially improving accuracy and reducing misclassifications.

A Contribution of each member

1. Rohan Regar: Implemented various classifier, conducted analysis of performance, ideas in ResNet-50,HoG and some parameter tuning.
2. Arnava Srivastava: Implemented failure case analysis of all models. Prepared presentation and youtube Video.
3. Dheeraj Dhakar: Implemented dimensionality Reduction and its analysis. Prepared the project page for overall idea of project.
4. Buddhav garg: Implemented feature extraction using HoG. Prepared the final report on LaTex.
5. Harshit Barwar: Performed dataset analysis, pre-testing using classical ML models.