KNN Report

Introduction

In this assignment, we explored the CIFAR-10 dataset, which consists of 60,000 32x32x3 color images in 10 different classes. We applied the k-nearest neighbors (k-NN) algorithm for image classification, using Euclidean (L2) distance as the similarity metric. The main objectives were to determine the optimal value of k using 5-fold cross-validation and evaluate the performance of the k-NN classifier on the testing data.

Data Preparation

The CIFAR-10 dataset was downloaded and extracted, resulting in 50,000 training images and 10,000 testing images. Due to memory limitations, a subset of the data was used, with 8,000 training examples and 1,000 testing examples. The images were flattened into vectors to be compatible with the k-NN classifier.

Model Building and Evaluation

We implemented the k-nearest neighbors algorithm, including methods to compute Euclidean distances and predict labels for testing data. Initially, we applied the classifier with k=5 to obtain a baseline accuracy on the test data. Subsequently, 5-fold cross-validation was performed to find the optimal value of k among the choices [1, 3, 5, 8, 10]. For each fold, the training data was divided into validation and training subsets, and the accuracy was computed for each k value. Finally, the best value of k was chosen based on the average accuracy across all folds, and the model was evaluated on the testing data using the selected k.

Results and Analysis

The accuracy of the k-NN classifier on the testing data with k=5 was found to be approximately 27.7%. Through cross-validation, we observed varying accuracies for different values of k across the folds. The average accuracies for k in the range [1, 3, 5, 8, 10] were 55.5%, 54.1%, 55.4%, 55.4%, and 55.8%, respectively. The best value of k selected based on cross-validation was 10, which resulted in an accuracy of approximately 27.6% on the testing data.

Data has apparently already been downloaded and unpacked.

Training data shape: (50000, 32, 32, 3)

Training labels shape: (50000,)

Test data shape: (10000, 32, 32, 3)

Test labels shape: (10000,)

(8000, 3072) (1000, 3072)

Got 277 / 1000 correct with k=5 => accuracy: 0.277000

Printing our 5-fold accuracies for varying values of k:

k = 1, accuracy = 0.273125

k = 1, accuracy = 0.277000

k = 1, accuracy = 0.280000

k = 1, accuracy = 0.277000

k = 1, accuracy = 0.288125

k = 1, accuracy = 0.277000

k = 1, accuracy = 0.268125

k = 1, accuracy = 0.277000

k = 1, accuracy = 0.278750

k = 1, accuracy = 0.277000

k = 3, accuracy = 0.260625

k = 3, accuracy = 0.277000

k = 3, accuracy = 0.274375

k = 3, accuracy = 0.277000

k = 3, accuracy = 0.276875

k = 3, accuracy = 0.277000

k = 3, accuracy = 0.244375

k = 3, accuracy = 0.277000

k = 3, accuracy = 0.261250

k = 3, accuracy = 0.277000

k = 5, accuracy = 0.281875

k = 5, accuracy = 0.277000

k = 5, accuracy = 0.276875

k = 5, accuracy = 0.277000

k = 5, accuracy = 0.288750

k = 5, accuracy = 0.277000

k = 5, accuracy = 0.269375

k = 5, accuracy = 0.277000

k = 5, accuracy = 0.266250

k = 5, accuracy = 0.277000

k = 8, accuracy = 0.284375

k = 8, accuracy = 0.277000

k = 8, accuracy = 0.280000

k = 8, accuracy = 0.277000

k = 8, accuracy = 0.286875

k = 8, accuracy = 0.277000

k = 8, accuracy = 0.265625

k = 8, accuracy = 0.277000

k = 8, accuracy = 0.266875

k = 8, accuracy = 0.277000

k = 10, accuracy = 0.280000

k = 10, accuracy = 0.277000

k = 10, accuracy = 0.290000

k = 10, accuracy = 0.277000

k = 10, accuracy = 0.291250

k = 10, accuracy = 0.277000

k = 10, accuracy = 0.282500

k = 10, accuracy = 0.277000

k = 10, accuracy = 0.262500

k = 10, accuracy = 0.277000

k = 1, avg. accuracy = 0.554625

k = 3, avg. accuracy = 0.540500

k = 5, avg. accuracy = 0.553625

k = 8, avg. accuracy = 0.553750

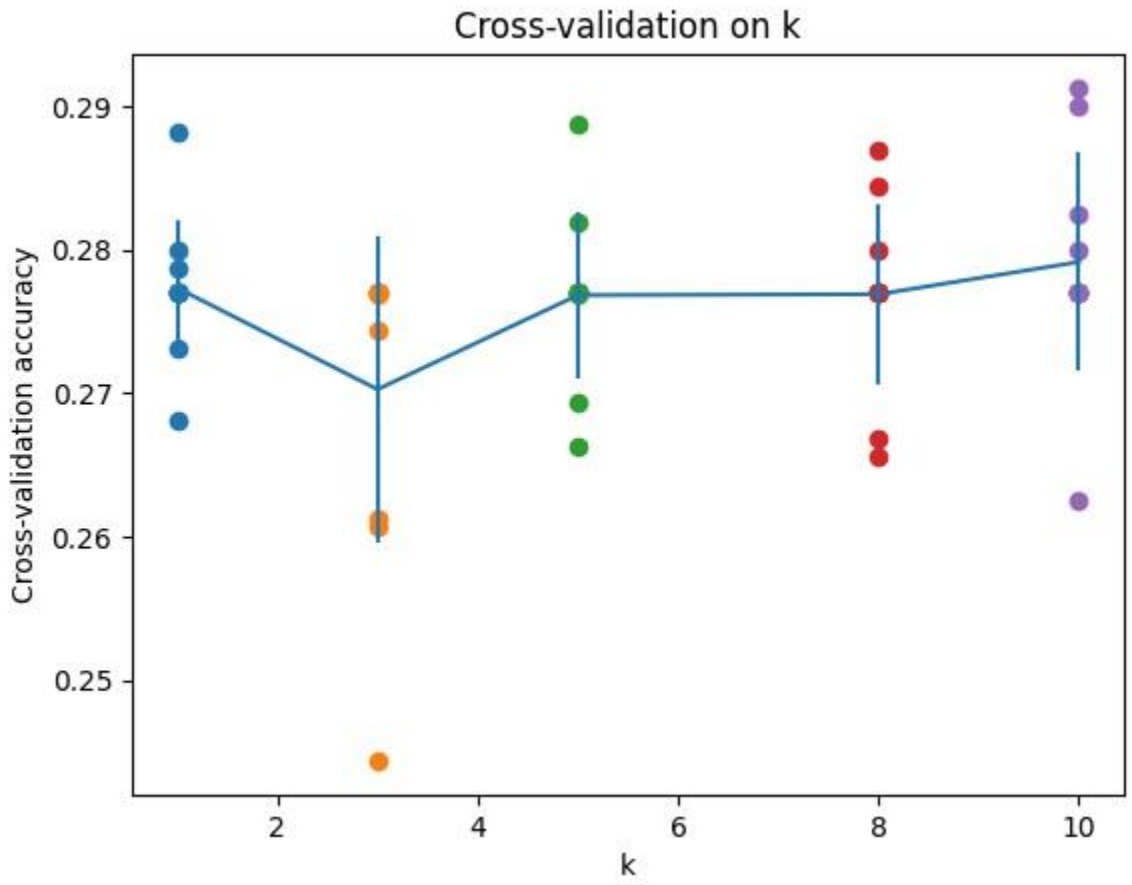
k = 10, avg. accuracy = 0.558250

Got 276 / 1000 correct on test data => accuracy: 0.276000

Best K value is: 10

Using Manhattan Distance Metric

Got 321 / 1000 correct with k=5 => accuracy: 0.321000



Discussion

Cross-validation was performed to find the optimal value of k because it provides a more robust evaluation of the model's performance. By splitting the training data into multiple folds and iterating over different values of k, we could assess the classifier's generalization ability across various subsets of the data. This approach helps prevent overfitting and ensures that the model's performance is representative.

Comparing the performance of the k-NN classifier to more sophisticated models such as neural networks (NN) or convolutional neural networks (CNN) on the CIFAR-10 dataset, we can observe several differences. While k-NN is simple and intuitive, it may not scale well to large datasets due to its computational complexity during inference. Additionally, k-NN lacks the ability to learn meaningful representations from the data, relying solely on similarity measures.

In contrast, NNs and CNNs can automatically learn hierarchical features from raw data, potentially achieving higher accuracies. These models can capture complex patterns and relationships in the data, making them more suitable for tasks like image classification. However, they require more computational resources for training and may suffer from overfitting if not properly regularized.

Manhattan Distance

We also experimented with Manhattan (L1) distance as an alternative distance metric for the k-NN classifier. Using Manhattan distance, the accuracy on the testing data with k=5 improved to approximately 32.1%. This indicates that different distance metrics may yield different results, and the choice of metric should be based on the characteristics of the data and the problem domain.