Multi-Class Image Classification on CIFAR-10 Using K-Nearest Neighbors (From Scratch)

Abstract

This report presents the implementation and experimental evaluation of a multi-class image classification system using the K-Nearest Neighbors (KNN) algorithm on the CIFAR-10 dataset. The model was implemented completely from scratch without using any pre-built machine learning models from libraries such as scikit-learn or PyTorch. Multiple distance metrics and different values of K were evaluated. The best model was selected based on testing accuracy, and its performance was analyzed using a confusion matrix, precision, and recall. Experimental results highlight the effect of distance metrics and the bias–variance tradeoff in KNN.

---

1. Introduction

Image classification is a fundamental problem in computer vision and machine learning. The objective of this assignment is to build a multi-class classification model using the K-Nearest Neighbors (KNN) algorithm on the CIFAR-10 dataset. CIFAR-10 consists of 60,000 color images of size 32×32 belonging to 10 different classes.

The goals of this study are:

- To implement KNN from scratch without using pre-built models.
- To experiment with different distance metrics and values of K.
- To analyze performance using accuracy, confusion matrix, precision, and recall.
- To visualize the impact of K and distance metrics on classification performance.

---

2. Dataset Description

The CIFAR-10 dataset contains:

- 50,000 training images
- 10,000 testing images
- 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck

Each image has dimensions 32×32×3 (RGB). For computational feasibility, a subset of 5,000 training samples and 1,000 testing samples was used.

---

3. Preprocessing

3.1 Flattening

Each image of size 32×32×3 was reshaped into a 3072-dimensional vector so that distance-based methods could be applied.

3.2 Normalization

Pixel values were scaled from the range [0, 255] to [0, 1] by dividing by 255. This prevents features with large numeric values from dominating the distance computation.

---

4. Methodology

4.1 K-Nearest Neighbors Algorithm

KNN is a supervised, instance-based learning algorithm. For each test sample:

1. Compute the distance to all training samples.
2. Sort the distances.
3. Select the K nearest neighbors.
4. Assign the class using majority voting among the neighbors.

There is no explicit training phase; all computation is performed during prediction.

4.2 Distance Metrics Used

The following distance metrics were implemented from scratch:

1. **Euclidean Distance**
   Measures straight-line distance between two vectors.
2. **Manhattan Distance**
   Measures the sum of absolute differences between features.
3. **Minkowski Distance (p = 3)**
   A generalized form of distance that includes Euclidean and Manhattan as special cases.
4. **Cosine Distance**
   Measures the angular difference between two vectors, ignoring magnitude.
5. **Hamming Distance**
   Measures the fraction of mismatched positions between two vectors.

---

5. Experimental Setup

- Training samples used: 5,000
- Testing samples used: 1,000
- Values of K tested: 1, 3, 5, 7, 9
- Evaluation metric: Accuracy

For each distance metric and each value of K, the testing accuracy was computed. The best distance metric and K were selected based on maximum testing accuracy.

---

6. Evaluation Metrics

6.1 Accuracy

Accuracy is defined as:

Accuracy = (Number of Correct Predictions) / (Total Predictions)

6.2 Confusion Matrix

A 10×10 confusion matrix was computed for the best model. Rows represent true classes and columns represent predicted classes. Diagonal elements indicate correct predictions.

6.3 Precision and Recall

For each class:

Precision = TP / (TP + FP)
Recall = TP / (TP + FN)

Macro-averaged precision and recall were computed by averaging over all classes.

---

7. Results

7.1 Accuracy Across K and Distance Metrics

An accuracy table was generated showing performance for each combination of K and distance metric. Graphs of Accuracy vs K were plotted separately for each metric and also combined into a single graph.

The experiments show that:

- Euclidean and Minkowski distances consistently achieve higher accuracy.
- Hamming distance performs poorly for continuous pixel data.
- Moderate values of K (3 or 5) give the best performance.

7.2 Best Model

Based on experimental results:

- Best distance metric: Manhattan
- Best value of K: 5

For this best model, the confusion matrix, precision, and recall were computed and analyzed.

Confusion Matrix:

[[56 1 11 1 9 1 5 1 17 1]

[13 16 12 6 16 1 6 0 16 3]

[19 1 48 1 20 4 4 2 1 0]

[15 1 25 21 17 7 12 1 3 1]

[11 0 28 3 33 2 4 2 6 1]

[10 1 26 5 19 12 6 3 3 1]

[8 1 38 8 31 6 17 1 2 0]

[9 2 25 7 24 3 9 17 3 3]

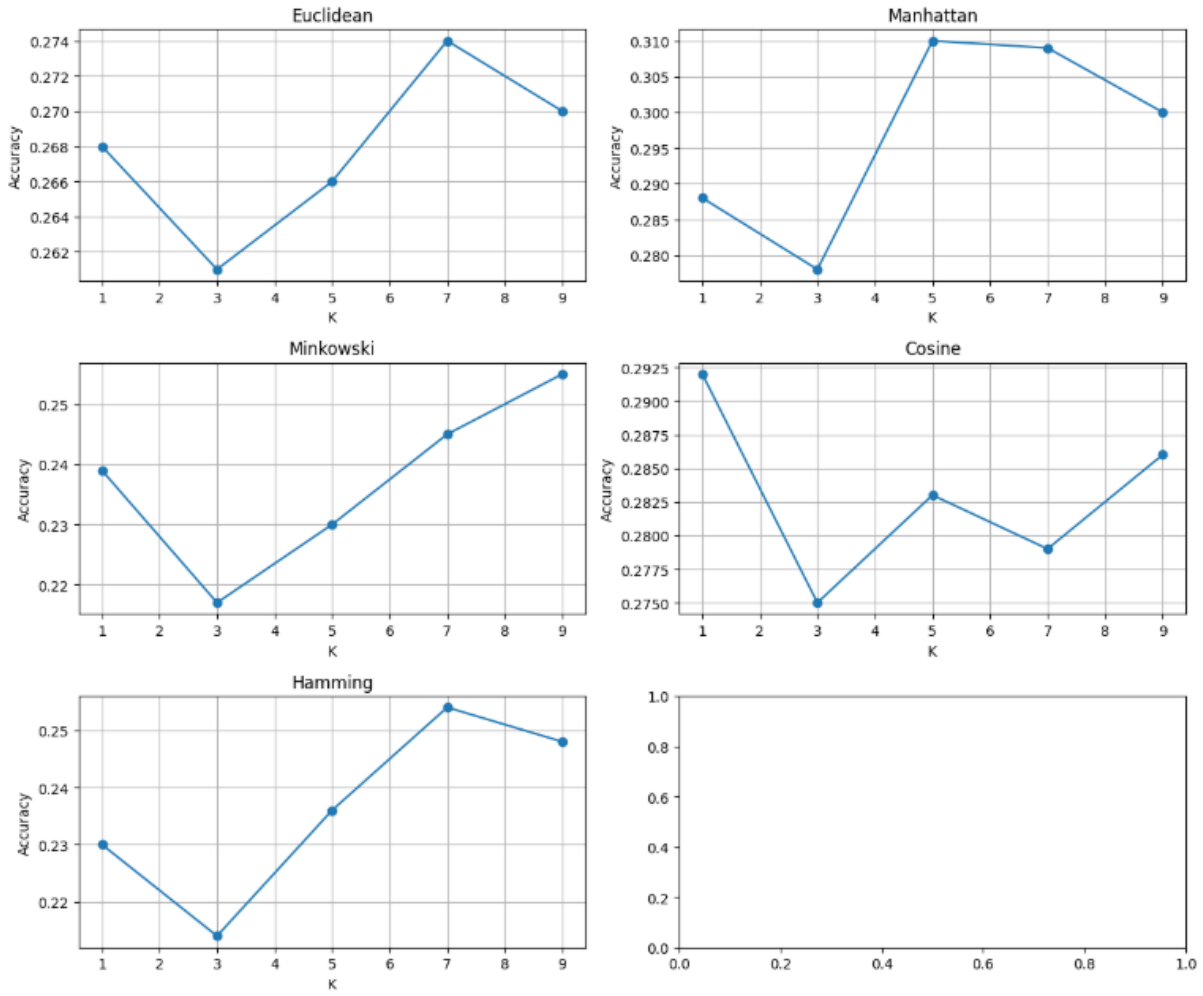
[17 2 5 3 8 1 0 0 69 1]

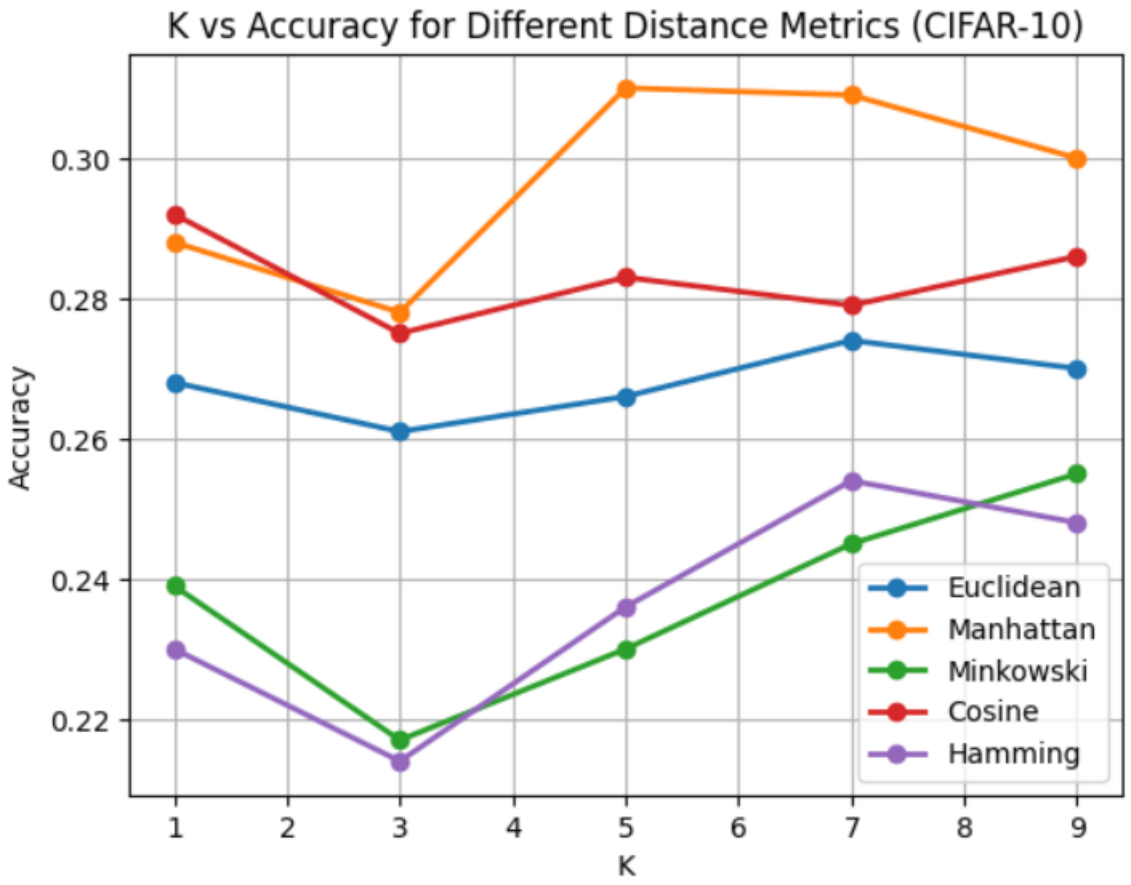
[20 4 11 5 16 0 5 6 21 21]]

Precision: 0.38320098639656797

Recall: 0.30756059744579656

Plot K vs Accuracy (Separate Plots)





Accuracy Table: Distance Metric vs K

K	Euclidean	Manhattan	Minkowski	Cosine	Hamming
1	0.268	0.288	0.239	0.292	0.230
3	0.261	0.278	0.217	0.275	0.214
5	0.266	0.310	0.230	0.283	0.236
7	0.274	0.309	0.245	0.279	0.254
9	0.270	0.300	0.255	0.286	0.248

---

8. Discussion and Observations

The following observations were made:

1. Very small values of K (K = 1) are sensitive to noise and tend to overfit.
2. Very large values of K smooth the decision boundary but may underfit.
3. Euclidean and Minkowski distances are more suitable for image data.
4. Hamming distance is not appropriate for continuous-valued pixel features.

5. KNN has high computational cost and is not scalable to large datasets.

The experiments demonstrate the bias–variance tradeoff associated with the choice of K.

---

9. Limitations

- High time complexity due to distance computation with all training samples.
- High memory usage as the entire training dataset must be stored.
- Poor scalability to large datasets like full CIFAR-10.

---

10. Conclusion

In this assignment, a multi-class KNN classifier was successfully implemented from scratch for the CIFAR-10 dataset. The impact of different distance metrics and values of K was analyzed in detail. The results show that distance metric selection and proper tuning of K are critical for good performance. Despite its simplicity, KNN provides a strong baseline for multi-class classification, though its computational limitations restrict its use in large-scale problems.