Neural Network and Deep Learning - Intermediate Assignment Report

Nearest-Neighbor vs Nearest Centroid

Athanasios Michalopoulos
Department of Electrical and Computer Engineering
Aristotle University of Thessaloniki

***Abstract*—In this project, two methods of classification are implemented and evaluated in terms of their classification accuracy. Two widely known classification algorithms are applied: the k-Nearest Neighbors (k-NN) algorithm, implemented with 1 and 3 neighbors, and the Nearest Centroid algorithm. Dataset CIFAR-10, containing images from 10 categories, has been selected, in order to evaluate the performance of the two methods in classifying an image into the correct object category.**

I. Introduction

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. Out of the total images, 50,000 are used for training and the remaining 10,000 for testing. An RGB image is made up of three distinct color channels — red, green, and blue. Each channel represents the intensity of its corresponding color using numerical values. These intensity levels range from 0 to 255, giving each channel 256 possible shades. In this color model, R stands for red, G for green, and B for blue, which are represented by the values (255, 0, 0), (0, 255, 0), and (0, 0, 255) respectively. The mathematical background of the algorithms will be examined in Section 2, followed by classification results, confusion matrices and conclusions derived from them, in Section 3.

II. Mathematical Background of Classification Algorithms

A. k-Nearest Neighbor

k-Nearest Neighbor (k-NN) classification method, is one of the most well known supervised machine learning algorithms used for classification and sometimes for regression. Consider a dataset where data points are labeled according to their classes. When a new test instance with an unknown category arrives, its Euclidean distance to all points in the dataset is calculated, as shown below:

$$d(\mathbf{x}_{\text{test}}, \mathbf{x}_i) = \sqrt{\sum_{j=1}^{p} \left(x_{\text{test},j} - x_{i,j}\right)^2} \tag{1}$$

, where

- $\mathbf{x}_{\text{test}} = (x_1, x_2, \ldots, x_p)$ is the feature vector of the new data point,
- $\mathbf{x}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,p})$ is the feature vector of a point in the training set,
- p is the number of features (dimensions),
- j is the index of the feature dimension

As soon as all distances are calculated, the algorithm identifies the k-nearest neighbors, meaning k-samples with the minimum distance from test instance. The algorithm then defines the class of the test sample, according to the majority class of these k-nearest samples. For example, based on the CIFAR-10 dataset, in the case of the 3-NN algorithm implementation, if two out of the three closest samples belong to the class dog and the remaining one belongs to the class cat, then the test sample is classified as a dog.

B. Nearest Centroid

The Nearest Centroid Classifier (NCC), is another supervised machine learning algorithm. At first, the geometric centre, or centroid (representing the mean feature vector of the class), of all data points in a class is calculated as shown below:

$$\mathbf{c}_j = \frac{1}{|S_j|} \sum_{\mathbf{x}_i \in S_j} \mathbf{x}_i \tag{2}$$

where:

- $\mathbf{c}_j$ is the centroid (mean vector) of class j,
- S_j is the set of all data points belonging to class j,
- $|S_j|$ is the number of data points (cardinality) in class j,
- $\mathbf{x}_i$ is an individual feature vector (data point) in the dataset.

Then, the Euclidean distance from a new test sample to the centroid of all classes is calculated. The sample is finally classified by identifying and selecting the class of the nearest centroid. For example, based on the CIFAR-10 dataset, if the distance between the test point and the centroid of class dog is the smallest of all distances, then the new sample is classified as dog.

In order to enhance the algorithm's efficiency, we implemented pixel normalization or PCA transformation. Each case is examined seperately. By normalizing (each pixel gets a value between 0 and 1, instead of 0–255), we standardize the data, making the algorithm's distance calculations fair and accurate.

By implementing PCA transformation, redundant and noisy information is eliminated, drastically cutting down the computational cost without significant loss of relevant data. This is achieved due to PCA's ability to identify and discard the dimensions that contribute the least to the overall variance. As a result, the algorithm focuses on the most informative features, improving its performance, while decreasing the time demanded for the task. By setting PCA to 100, the data is

reduced to 100 principal components—representing the most informative features out of the original 3,072—while preserving as much of the training set's variance as possible($\approx 90\%$).

III. Conclusions

In this section, the performance of the classifiers is summarized in the following table, followed by the confusion matrix comparisons. It is evident that the highest classification accuracy is achieved when using the k-NN algorithm with k=1 combined with PCA. It can also be observed that pixel normalization did not lead to any enhancement in classification accuracy. Moreover, the k-NN algorithm consistently outperforms the NCC classifier in terms of accuracy. Finally, it should be noted that PCA slightly improved accuracy in both algorithms, as predicted by theory. These conclusions can be drawn by analyzing the confusion matrices presented below the table, as well. More specifically, the diagonal entries of the confusion matrices, representing correct classifications, are moderate when using k-NN without PCA. After applying PCA, the diagonal values in the confusion matrices increase substantially, often exceeding 500 for certain classes — with ship and airplane being the most accurately predicted. Notably, k-NN with $k = 1$ combined with PCA achieves the highest number of correct predictions along the diagonal, indicating the best performance. Moreover, when using the NC classifier, the confusion matrix shows misclassifications across most classes, with lower diagonal values compared to those of the k-NN models. Even after applying PCA, the pattern doesn't change much, there's even a small decline in the diagonal for a few classes. For Nearest Centroid algorithm, airplane and frog are the two most correct-predicted classes.

TABLE I: Performance summary across 10 CIFAR-10 classes

Classification algorithm	Classification Accuracy
k-NN with k=1	35,39
k-NN with k=1 and normalization	35,39
k-NN with k=1 and PCA	38,53
k-NN with k=3	33,03
k-NN with k=3 and normalization	33,03
k-NN with k=3 and PCA	36,39
NC	27,74
NC and normalization	27,74
NC and PCA	27,66

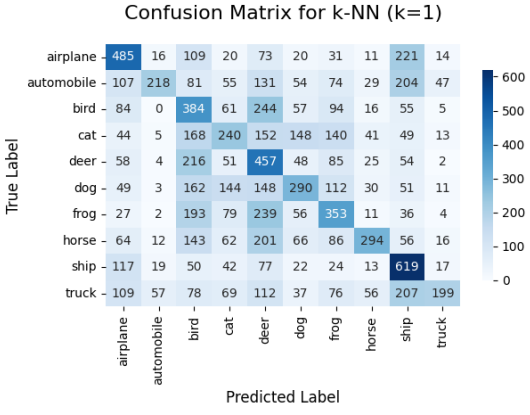

Figure 1: k-NN with k=1

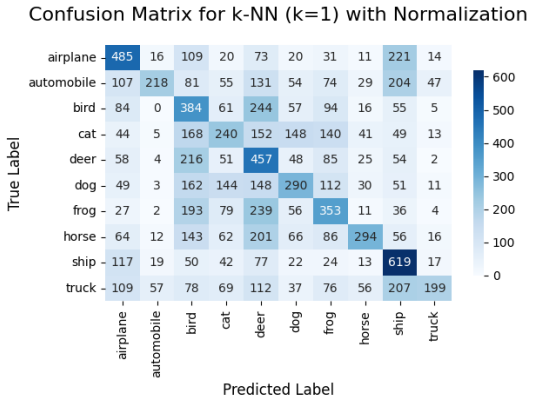

Figure 2: k-NN with k=1 and normalization

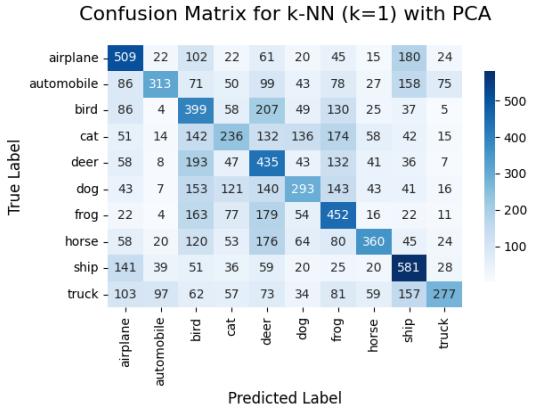

Figure 3: k-NN with k=1 and PCA

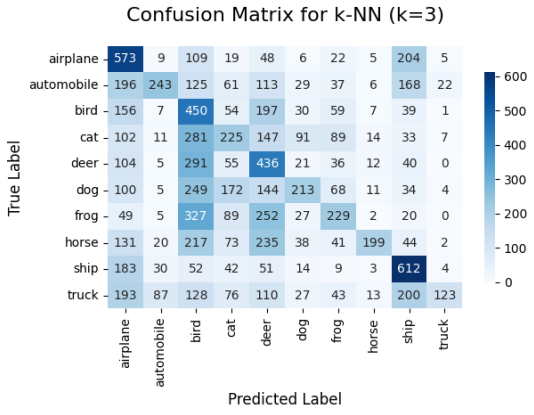

Figure 4: k-NN with k=3

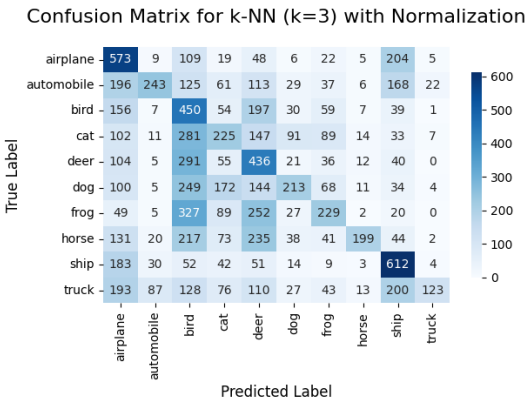

Figure 5: k-NN with k=3 and normalization

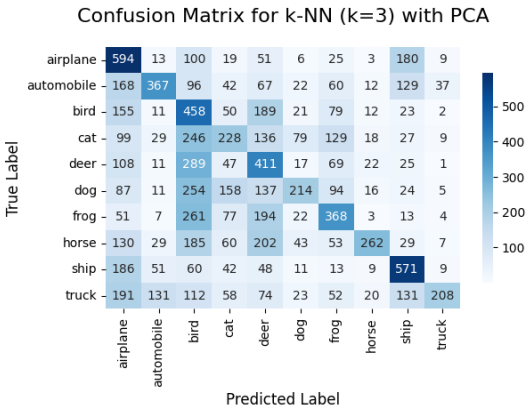

Figure 6: k-NN with k=3 and PCA

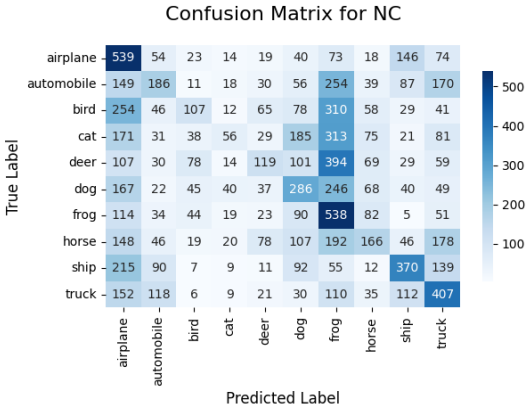

Figure 7: Nearest Centroid

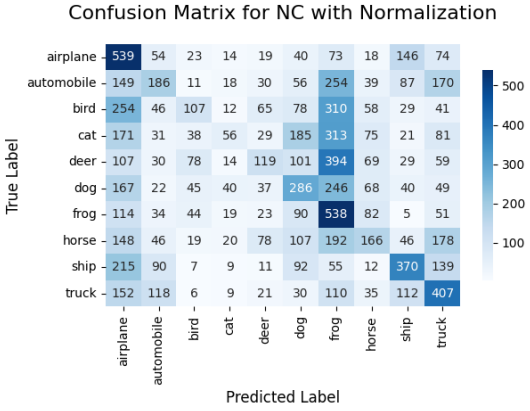

Figure 8: Nearest Centroid with normalization

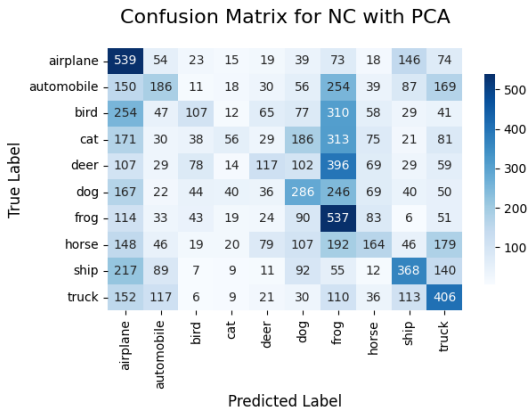

Figure 9: Nearest Centroid with PCA

REFERENCES

[1] Alex Krizhevsky.(2009). CIFAR-10 and CIFAR-100 datasets: https://www.cs.toronto.edu/ kriz/cifar.html.