

Graph-Based Nearest Centroid for Image Classification

Método do Centróide mais Próximo Baseado em Grafos para Classificação de Imagens

Alexandre L. M. Levada^{1*}

Abstract: Image classification is a fundamental task for pattern recognition and computer vision. Many different classifiers have been proposed to overcome the limitations of supervised learning in practical problems. Despite all the efforts, due to the curse of the dimensionality, classifying high-dimensional data still pose a challenge to researchers and practitioners from several fields of science. In this paper, we propose a graph-based nearest centroid classifier (Graph-NCC), method for supervised classification that builds a discrete approximation to the data manifold and employs shortest paths to approximate the geodesic distances between sample points. An analysis of the complexity reveals that Graph-NCC is log-linear in the number of samples and linear in the number of edges, showing that the method is quite efficient in comparison to modern supervised classification techniques. We performed computational experiments with real image datasets to demonstrate the effectiveness of Graph-NCC. The obtained results show that the proposed Graph-NCC algorithm is capable of improving the balanced accuracy in comparison to the regular nearest neighbor classifier (NCC). In some cases, the proposed method also overperforms classical machine learning methods, as support vector machines (SVM), k-nearest neighbors classifier (k-NN) and XGBoost.

Keywords: image classification — nearest centroid classifier — graph-based learning — geodesic distances

Resumo: A classificação de imagens é uma tarefa fundamental para o reconhecimento de padrões e a visão computacional. Diversos classificadores diferentes foram propostos para superar as limitações do aprendizado supervisionado em problemas práticos. Apesar de todos os esforços, devido à maldição da dimensionalidade, a classificação de dados de alta dimensionalidade ainda representa um desafio para pesquisadores e profissionais de diversas áreas da ciência. Neste artigo, propomos um classificador de centróide mais próximo baseado em grafos, um método para classificação supervisionada que constrói uma aproximação discreta para a variedade de dados e emprega caminhos mínimos para aproximar as distâncias geodésicas entre pontos amostrais. Uma análise da complexidade revela que o Graph-NCC é log-linear no número de amostras e linear no número de arestas, mostrando que o método é bastante eficiente em comparação com técnicas modernas de classificação supervisionada. Realizamos experimentos computacionais com conjuntos de dados de imagens reais para demonstrar a eficácia do Graph-NCC. Os resultados obtidos mostram que o algoritmo Graph-NCC proposto é capaz de melhorar a precisão balanceada em comparação com o classificador de vizinho mais próximo (NCC) regular. Em alguns casos, o método proposto também supera métodos clássicos de aprendizado de máquina, como as máquinas de vetores de suporte (SVM), classificador de k-vizinhos mais próximos (k-NN) e XGBoost.

Palavras-Chave: classificação de imagens — classificador do centróide mais próximo — aprendizado em grafos — distâncias geodésicas

¹ Departamento de Computação, Universidade Federal de São Carlos (UFSCar), Brazil

*Corresponding author: alexandre.levada@ufscar.br

DOI: <http://dx.doi.org/10.22456/2175-2745.150723> • Received: 06/10/2025 • Accepted: 26/10/2025

CC BY-NC-ND 4.0 - This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

1. Introduction

Supervised image classification is a fundamental application in machine learning, as it plays a crucial role in several problems, including those related to data science, pattern recognition, and image processing [1, 2, 3, 4, 5]. In supervised learning, the algorithm is trained on a labeled dataset, where each input point is associated with a corresponding output

label. The goal is for the algorithm to learn a mapping from inputs to outputs, allowing it to make predictions on unseen data. Among the main applications of supervised classification, we can mention predictive modeling, neural networks, the development of new performance metrics, adaptability, and generalization in concept drift problems [6, 7, 8].

One of the earliest and simplest supervised classifiers described in the literature is the nearest centroid classifier (*Nearest Centroid Classifier* or NCC). It is a simple and intuitive algorithm belonging to the category of prototype-based classifiers, where class prototypes or centroids are used to represent each class [9]. NCC is particularly effective in situations where class distributions are approximately Gaussian and class centroids are reliable representatives. Its main limitations are [10]:

1. sensitivity to outliers, since the presence of a single one can affect the position of the centroid (mean);
2. NCC assumes that the covariance matrices of all classes are equal, i.e., the classes are circularly symmetric, which is not a realistic assumption;
3. NCC is a linear classifier and may struggle to capture complex decision boundaries in high-dimensional spaces;
4. limited discriminative power, as it is strongly tied to Euclidean distance, which is known to be an inadequate metric in high-dimensional spaces, such as in image classification problems.

A critical challenge in classifying high-dimensional data is the curse of dimensionality, which severely degrades the discriminative power of Euclidean distances. Theoretically, as the dimensionality d increases, the ratio between the distance to the nearest neighbor and the farthest neighbor converges to 1, which means that all pairwise distances become asymptotically equal [11]. This phenomenon, known as distance concentration, renders Euclidean metrics ineffective for distinguishing between samples in high-dimensional spaces [12].

To overcome some of these limitations, in this paper we propose the Graph-NCC, a graph-based classifier that uses a discrete approximation to the data manifold to replace Euclidean distances with geodesic distances, computed as the lengths of shortest paths. With this modification, the proposed method has two main contributions:

- the replacement of Euclidean distances by geodesic distances makes it possible to construct more complex decision boundaries, since the Graph-NCC is able to handle non-spherical classes (it does not assume that covariance matrices are proportional to the identity matrix);
- Graph-NCC does not suffer from the loss of discriminative power of Euclidean distance in high-dimensional spaces, which leads to better results in cases where the number of features d is much larger than the number of samples n , mitigating the curse of dimensionality.

To evaluate the proposed method, we compared it with the standard NCC and with the SVM, k-NN and XGBoost classifiers on image classification problems.

The remainder of the paper is organized as follows: Section 2 presents an overview of the nearest centroid classifier and describes the proposed Graph-NCC algorithm, explaining in detail how it works and analyzing its computational complexity. Section 3 shows the computational experiments with datasets composed of real high-dimensional images in terms of quantitative measures. Finally, Section 4 presents the conclusions and final considerations.

2. Materials and Methods

The supervised classification problem assumes that we have a tuple $Z = (X, Y)$, such that $X = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$ is a set of n samples with $\vec{x}_i \in \mathbb{R}^d$, and $Y = \{y_1, y_2, \dots, y_n\}$ is the set of labels with $y_i \in \{1, 2, \dots, c\}$, where c is the number of classes in the problem. A classifier is a function $f: X \rightarrow Y$ that predicts a label $y_i \in Y$ for every $\vec{x}_i \in X$. In geometric terms, the classifier f induces a partition of X into classes.

2.1 The Nearest Centroid Classifier

The nearest centroid classifier (NCC) is a simple and intuitive supervised learning algorithm used for pattern recognition and classification. Basically, the decision rule for NCC is: assign the sample $\vec{x}$ to the class whose centroid (mean vector) is the closest. Mathematically, we have:

$$y = \arg \min \{D_j(\vec{x})\} = \arg \min \{\|\vec{x} - \vec{\mu}_j\|^2\} \quad (1)$$

for $j = 1, 2, \dots, c$, where $D_j(\vec{x})$ denotes the Euclidean distance from sample $\vec{x}$ to the centroid of class j . Algorithm 1 shows the pseudocode for the training phase of the nearest centroid classifier. Here, the input parameters are X , the training data matrix of size $n \times d$; Y , the label vector of size $n \times 1$; n , the number of samples; and c , the number of classes. The output is the c centroids, one for each class. The algorithm iterates over each class, identifies the samples, and computes the mean vectors (centroids).

Algorithm 1 NCC training algorithm

```

function TRAIN_NCC( $X, Y, n, c$ )
  for  $j \leftarrow 1; j \leq c; j++$  do
     $k \leftarrow \text{where}(Y == j)$ 
     $n_j \leftarrow \text{len}(k)$ 
     $Z \leftarrow X[k, :]$ 
     $\vec{\mu}_j \leftarrow \frac{1}{n_j} \sum_{i=1}^{n_j} Z[i, :]$ 
  end for
  return  $\vec{\mu}_1, \vec{\mu}_2, \dots, \vec{\mu}_c$ 
end function

```

It can be observed that the computational complexity of the NCC training algorithm is $O(cnd)$, where c is the number of classes, n is the number of samples in the training set, and d is the number of features. Algorithm 2 shows the pseudocode for the testing phase of the nearest centroid classifier. In

summary, for each test sample, the Euclidean distance to all centroids is computed, and then the label of the nearest centroid is assigned.

Algorithm 2 NCC testing algorithm

```

function TEST_NCC( $X, n, c, \bar{\mu}_1, \bar{\mu}_2, \dots, \bar{\mu}_c$ )
  for  $i \leftarrow 1; i \leq n; i++$  do
    for  $j \leftarrow 1; j \leq c; j++$  do
       $D_j \leftarrow \|\bar{x}_i - \bar{\mu}_j\|^2$ 
    end for
     $y_i \leftarrow \text{argmin}(\{D_1, D_2, \dots, D_c\})$ 
  end for
  return  $y_1, y_2, \dots, y_n$ 
end function

```

It is straightforward to note that the computational complexity of the NCC testing algorithm is $O(cnd)$, where c is the number of classes, n is the number of samples in the test set, and d is the number of features.

2.2 The Graph-based NCC Classifier

To overcome some limitations of the NCC classifier, this work proposes the graph-based nearest centroid classifier (Graph-NCC), replacing the Euclidean distance with approximate geodesic distances on the data manifold. This is possible due to the construction of a k -NN graph, which is a discrete way of approximating the unknown manifold structure. In summary, there are two direct contributions compared with the regular NCC: (1) it mitigates the NCC's inability to deal with non-spherical classes; and (2) it alleviates the negative effects of the curse of dimensionality caused by the sparsity of the feature space, especially in situations where the number of features d is high.

An important question at this point is: how can we compute approximate geodesic distances between points in a computationally efficient manner? One solution is the application of Dijkstra's algorithm. Basically, Dijkstra's algorithm employs dynamic programming and a greedy strategy to construct shortest paths in weighted graphs $G = (V, E, w)$ in a computationally efficient way, where V denotes the set of vertices, E denotes the set of edges, and $w : E \rightarrow \mathbb{R}^+$ is a weighting function. Algorithm 3 shows the pseudocode of Dijkstra's algorithm. First, we need to define the main variables used by the algorithm: $\lambda(v)$ is the length of the shortest path from the root s to vertex v ; $\pi(v)$ is the predecessor of vertex v in the shortest-path tree; and Q is the priority queue to store the vertices (the smaller $\lambda(v)$, the higher the priority).

It can be shown that the computational complexity of Dijkstra's algorithm is $O(m \log n)$ [13], where n and m denote the number of vertices and edges of G , respectively. A natural question arises: can Dijkstra's algorithm recover the geodesic distances in all weighted graphs? The following theorem shows that, if all edge weights are positive (as is the case in Graph-NCC), the answer is affirmative [13].

Algorithm 3 Dijkstra's algorithm

```

function DIJKSTRA( $G, w, s$ )
  for each  $v \in V$  do ▷ Initialize  $\lambda(v)$  and  $\pi(v)$ 
     $\lambda(v) \leftarrow \infty$ 
     $\pi(v) \leftarrow \text{nil}$ 
  end for
   $\lambda(s) \leftarrow 0$  ▷ Seed has  $\lambda(s) = 0$ 
   $S \leftarrow \emptyset$ 
   $Q \leftarrow \emptyset$ 
  for each  $v \in V$  do ▷ Insert vertices in  $Q$ 
    Insert( $Q, v$ )
  end for
  while  $Q \neq \emptyset$  do ▷ main loop
     $u \leftarrow \text{ExtractMin}(Q)$  ▷ Vertex with smallest  $\lambda$ 
     $S \leftarrow S \cup \{u\}$ 
    for each  $v \in N(u)$  do ▷ Process each neighbor
       $\lambda(v) \leftarrow \min\{\lambda(v), \lambda(u) + w(u, v)\}$ 
      if  $\lambda(v)$  was updated then ▷ Found better route
         $\pi(v) \leftarrow u$  ▷ Update  $v$ 's predecessor
        Decrease_Key( $Q, v, \lambda(v)$ )
      end if
    end for
  end while
end function

```

Theorem 1 Given a graph $G = (V, E, w)$ with $w : E \rightarrow \mathbb{R}^+$, Dijkstra's algorithm always terminates with $\lambda(v) = d(s, v)$ for all $v \in V$, where $d(s, v)$ denotes the geodesic distance from s to v .

Recall that $\lambda(v) \geq d(s, v)$ for all $v \in V$. Suppose vertex u is the first vertex removed from Q such that $\lambda(u) \neq d(s, u)$. Note that u cannot be the root of the tree, since $\lambda(s) = 0 = d(s, s)$. Furthermore, there must exist a path P_{su} ; otherwise, $\lambda(u) = \infty = d(s, u)$. Therefore, there exists a shortest path P_{su}^* . Prior to u being removed from Q , the path P_{su}^* contains $s \in S$ and $u \in V - S$. Let y be the first vertex on P_{su}^* such that $y \in V - S$ and $x \in S$ its predecessor. Since $x \in S$, we have $\lambda(x) = d(s, x)$. When x was removed from Q , the edge (x, y) was relaxed, yielding:

$$\lambda(y) = \lambda(x) + w(x, y) = d(s, x) + w(x, y) = d(s, y) \quad (2)$$

As y precedes u in P_{su}^* and all edge weights are non-negative, it follows that $d(s, y) \leq d(s, u)$, implying

$$\lambda(y) = d(s, y) \leq d(s, u) \leq \lambda(u). \quad (3)$$

However, since u is selected to leave Q , we also have $\lambda(u) \leq \lambda(y)$. Combining these inequalities gives $\lambda(u) = \lambda(y)$, which leads to

$$\lambda(y) = d(s, y) = d(s, u) = \lambda(u), \quad (4)$$

contradicting the assumption that $\lambda(u) \neq d(s, u)$ when u leaves Q , and thus completing the proof.

The graph-based nearest centroid classifier is an algorithm that constructs a k-NN graph to approximate the underlying data manifold in the input space. In machine learning, the manifold hypothesis posits that high-dimensional data often lie on a low-dimensional, hidden, nonlinear geometric structure embedded within the observed ambient space [14, 15]. A natural question that arises is: how accurately do optimal paths in k-NN graphs approximate the true geodesic distances on the data manifold? The Asymptotic Convergence Theorem establishes that, under certain conditions, the lengths of optimal paths in k-NN graphs, denoted by $d_G(\vec{x}_i, \vec{x}_j)$, converge to the geodesic distances $d_M(\vec{x}_i, \vec{x}_j)$ [16].

In essence, the authors demonstrate that the two metrics, $d_G(\vec{x}_i, \vec{x}_j)$ and $d_M(\vec{x}_i, \vec{x}_j)$, become arbitrarily close as the density of points increases [17]. In summary, the Asymptotic Convergence Theorem guarantees that, as the number of samples grows and k-NN graphs are constructed, the shortest paths in these graphs converge to the true geodesic distances on the hidden manifold. This theoretical result provides a rigorous justification for replacing Euclidean distances with geodesic distances in methods such as Graph-NCC.

Theorem 2 (Asymptotic Convergence Theorem) *Given parameters $\lambda_1, \lambda_2, \mu > 0$, as small as desired, then for a sufficiently high point density, the following inequality holds:*

$$1 - \lambda_1 \leq \frac{d_G(\vec{x}_i, \vec{x}_j)}{d_M(\vec{x}_i, \vec{x}_j)} \leq 1 + \lambda_2 \quad (5)$$

with probability $1 - \mu$, where $d_G(\vec{x}_i, \vec{x}_j)$ denotes the distance measured on the k-NN graph, and $d_M(\vec{x}_i, \vec{x}_j)$ denotes the unknown geodesic distance on the manifold.

Note that the training of the Graph-NCC is identical to that of the regular NCC, which means that the proposed method is a lazy classifier; most of the computational cost occurs during the testing phase. Algorithm 4 presents the pseudocode for the testing phase of the graph-based nearest centroid classifier. The input parameters are: the data matrix X , the number of samples n , the number of neighbors k , the number of classes c , and the c centroids obtained during the training phase.

In summary, the algorithm first constructs a k-NN graph from the test samples, using Euclidean distance to weight the edges. It then computes the geodesic distances to the c centroids using Dijkstra's algorithm for each test sample, and finally assigns the label corresponding to the centroid with the minimum geodesic distance. Note that the computation of the minimum spanning tree ensures that the k-NN graph is necessarily connected.

The computational complexity of the graph-based nearest centroid classifier is a function that depends directly or indirectly on the following quantities: the number of samples n , the number of features d , the number of clusters c , the number of neighbors in the k-NN graph k , and the number of

Algorithm 4 Graph-based NCC testing algorithm

```

function TEST_GEONCC( $X, n, k, c, \vec{\mu}_1, \vec{\mu}_2, \dots, \vec{\mu}_c$ )
   $K_n \leftarrow \text{Complete\_Graph}(X, n - 1)$     ▷ Complete graph
   $T \leftarrow \text{MST}(K_n)$                     ▷ Minimum spanning tree
   $G \leftarrow \text{kNN\_Graph}(X, k)$            ▷ Construct k-NN graph
   $G \leftarrow G + T$                        ▷ Ensure  $G$  is connected
  for  $i \leftarrow 1; i \leq n; i++$  do
     $D_1, D_2, \dots, D_c \leftarrow \text{Dijkstra}(G, \vec{x}_i)$ 
     $y_i \leftarrow \text{argmin}(\{D_1, D_2, \dots, D_c\})$ 
  end for
  return  $y_1, y_2, \dots, y_n$ 
end function

```

edges in the k-NN graph m . Note that the k-NN graph can be constructed in $O(nd \log n)$ using a KD-Tree, which is the data structure employed by most Python libraries, such as scikit-learn [18]. Prim's algorithm is executed once to extract the MST, resulting in a cost of $O(m \log n)$. Dijkstra's algorithm is executed n times, resulting in a total cost of $O(nm \log n)$. Therefore, considering the dominant terms, the total cost of the Graph-NCC algorithm is $O(nd \log n) + O(nm \log n)$, indicating a log-linear complexity in the number of vertices and a linear complexity in the number of edges.

3. Experiments and Results

The computational experiments were designed to evaluate the performance of the proposed Graph-NCC algorithm in comparison with standard NCC, k-NN, SVM, and XGBoost classifiers on various image classification tasks. The following datasets were used:

- **digits:** *Pen-based recognition of handwritten digits.* This dataset contains $n = 1797$ images of size 8×8 , resulting in $m = 64$ features, organized into $c = 10$ classes [19].
- **mfeat-pixel:** one of six datasets capturing features of handwritten digits extracted from a collection of Dutch utility maps. It comprises $n = 2000$ samples with $m = 15 \times 16 = 240$ features, distributed across $c = 10$ classes [20].
- **optdigits:** *Optical recognition of handwritten digits.* The original $n = 5620$ bitmaps of size 32×32 are divided into non-overlapping 4×4 blocks, and the number of pixels in each block is counted to produce input matrices of size 8×8 , yielding $m = 64$ features. The samples are grouped into $c = 10$ classes [21].
- **satimage:** this dataset contains multispectral pixel values in 3×3 neighborhoods from a satellite image. The task is to predict the label of the central pixel. The dataset covers a small region of a Landsat satellite image (82×100 pixels), totaling $n = 6430$ samples, each

with $m = 36$ features ($9 \text{ pixels} \times 4 \text{ bands}$), distributed across $c = 6$ classes [22].

- **texture:** this dataset comprises $c = 11$ classes, each containing 500 instances, totaling $n = 5500$ samples with $m = 40$ features. Each class represents a distinct texture type from the Brodatz texture album [23].
- **one-hundred-plants-margin:** the dataset contains $m = 64$ boundary features directly extracted from $n = 1600$ binary images of $c = 100$ leaf specimens (16 samples for each class) [24].
- **Olivetti_Faces:** this dataset contains ten distinct images for each of $c = 40$ individuals, totaling $n = 400$ samples. The original images were 92×112 , while the available version contains 64×64 images, resulting in $m = 4096$ features [25].
- **Satellite:** this dataset consists of features extracted from satellite images. The goal is to classify regions based on land type: red soil, gray soil, wet gray soil, and very wet gray soil as the normal class. Two semantically different classes - cotton crop and soil with plant residue - are treated as anomalies, defining a $c = 2$ classification problem. After merging the original training and test sets, the resulting dataset contains 5,025 normal instances and 75 randomly sampled anomalies (1.49%), with $m = 36$ features, for a total of $n = 5100$ samples.

All eight datasets are publicly available at (www.openml.org) and can be downloaded with Python's scikit-learn library function `fetch_openml` from the datasets package.

In the first experiment, the goal is to compare regular NCC and Graph-NCC in image classification problems, using as metric the balanced accuracy. To perform a robust analysis, we propose the following experimental protocol: 1) Evaluation strategy: to assess stability and statistical significance, we conducted experiments in which, for fixed training sizes (10%, 20%, 30%, 40% and 50%), we generated 30 independent random partitions (training/testing) for each size; 2) Robust statistical analysis: for each partition, we will run both classifiers (regular NCC and Graph-NCC), which will allow us to legitimately calculate means, standard deviations and confidence intervals based on the variation in the training samples; 3) Significance tests: we will include statistical tests (paired t-test) to verify the significance of performance differences. As the primary objective was to evaluate the ability of the proposed method to learn from less samples, we used reduced training sets. The obtained results are shown in Tables 1 and 2. According to a paired t-test, the differences in the performances are statistically significant for $\alpha = 0.05$ ($p\text{-value} = 6.67 \times 10^{-8}$), showing the superiority of Graph-NCC in these image datasets. Figure 1 shows the average balanced accuracy curves for the digits, mfeat-pixel, optdigits, satimage, and texture datasets, once again showing that Graph-NCC outperforms regular NCC in all training set sizes.

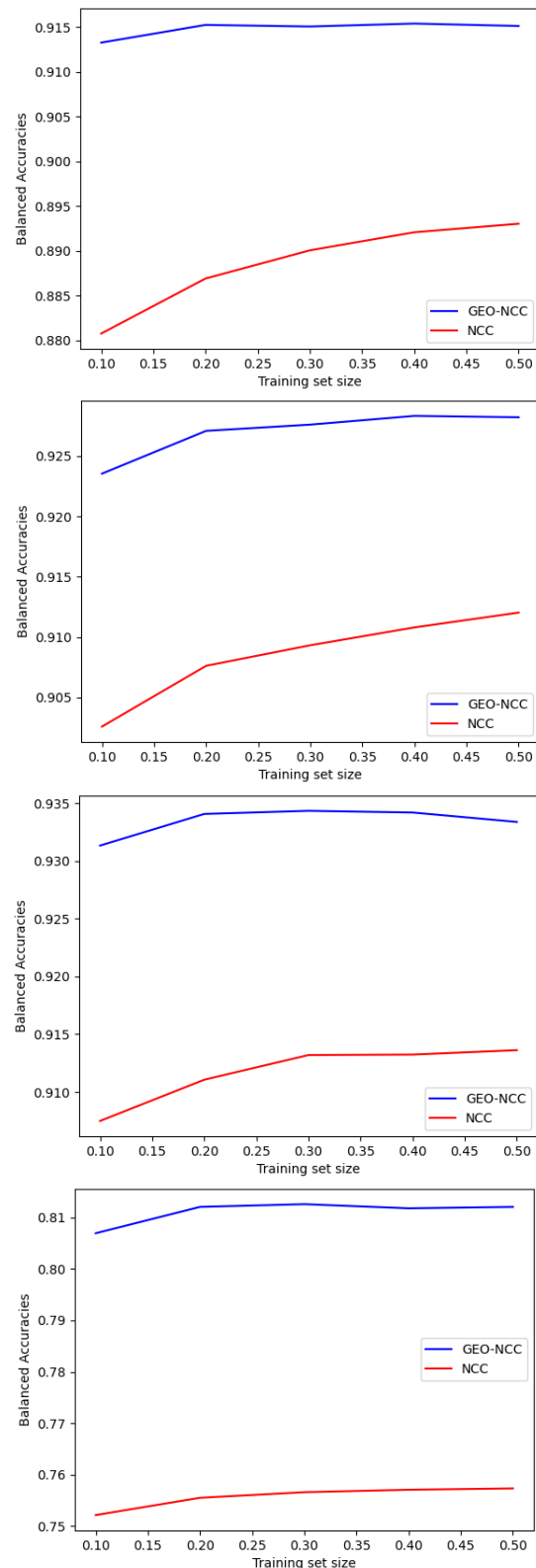


Figure 1. Average balanced accuracies for standard NCC and Graph-NCC across various training/testing partitions.

Table 1. Average balanced accuracies and standard deviations for regular NCC for different training set sizes, from 10% to 50% of samples (for each training set size, 30 rounds using random sampling are performed).

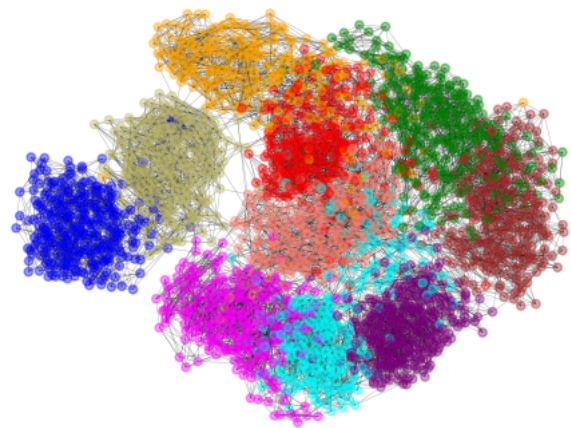
Dataset	Training set sizes for regular NCC				
	10%	20%	30%	40%	50%
digits	0.8887 ± 0.0094	0.8869 ± 0.0111	0.8900 ± 0.0113	0.8920 ± 0.0111	0.8930 ± 0.0106
mfeat-pixel	0.9025 ± 0.0084	0.9073 ± 0.0093	0.9093 ± 0.0086	0.9108 ± 0.0085	0.9120 ± 0.0086
optdigits	0.9074 ± 0.0066	0.9110 ± 0.0067	0.9131 ± 0.0067	0.9132 ± 0.0070	0.9136 ± 0.0067
satimage	0.7596 ± 0.0081	0.7638 ± 0.0082	0.7659 ± 0.0079	0.7668 ± 0.0079	0.7671 ± 0.0077
texture	0.7521 ± 0.0145	0.7554 ± 0.0118	0.7565 ± 0.0115	0.7570 ± 0.0110	0.7573 ± 0.0108

Table 2. Average balanced accuracies and standard deviations for Graph-NCC for different training set sizes, from 10% to 50% of samples (for each training set size, 30 rounds using random sampling are performed).

Dataset	Training set sizes for Graph-NCC				
	10%	20%	30%	40%	50%
digits	0.9132 ± 0.0087	0.9152 ± 0.0083	0.9150 ± 0.0081	0.9154 ± 0.0090	0.9151 ± 0.0095
mfeat-pixel	0.9345 ± 0.0091	0.9235 ± 0.0090	0.9271 ± 0.0083	0.9276 ± 0.0082	0.9283 ± 0.0078
optdigits	0.9313 ± 0.0051	0.9341 ± 0.0050	0.9343 ± 0.0055	0.9342 ± 0.0055	0.9339 ± 0.0059
satimage	0.7652 ± 0.0088	0.7682 ± 0.0086	0.7702 ± 0.0084	0.7711 ± 0.0091	0.7721 ± 0.0090
texture	0.8069 ± 0.0219	0.8120 ± 0.0182	0.8126 ± 0.0169	0.8117 ± 0.0157	0.8120 ± 0.0156

To better understand the behavior of the proposed geodesic nearest centroid classifier, we provide visualizations of the underlying k-NN graph constructed from the digits dataset. This graph serve a dual purpose. First, they reveal the local neighborhood structures that directly influence the geodesic distances used in our method, thereby offering an intuitive interpretation of how class centroids are connected through the data manifold. Second, they allow us to identify potential issues such as class overlap, noise, or irregular connectivity patterns that may affect classification performance. It complements the quantitative results by providing a qualitative assessment of how the graph topology supports the decision boundaries induced by the classifier and also shows up the role of the geodesic distances. Moreover, since our method explicitly relies on geodesic distances defined on these graphs, their visualization highlights the geometric rationale behind our approach, making the relation between the data distribution, the graph structure, and the classification outcome more transparent. Figure 2 shows the k-NN graph for the optdigits dataset labeled with the true labels.

In the second set of experiments, the primary objective is to compare the performance of Graph-NCC with k-NN, SVM (rbf kernel), and XGBoost classifiers on the Olivetti-Faces, Satellite and one-hundred-plants-margin image datasets in terms of balanced accuracy. Using a holdout 60%-40% strategy, we generated 30 independent random partitions (training/testing) and compute the average balanced accuracy. Moreover, the standard deviations are also computed to measure the variability of the scores. The quantitative results for this experiment are presented in Table 2. The results indicate that, in terms of balanced accuracy, Graph-NCC is able to achieve performance comparable to 1-NN, and superior to

**Figure 2.** k-NN graph of the optdigits dataset with the parameter number of neighbors $k = \log_2(n)$.

SVM, and XGBoost in these datasets, indicating that the proposed method represent a promising alternative for image classification problems.

In our experiments, we adopted two values of k in the k-NN classifier: $k = 1$ and $k = 5$. Although $k = 1$ often yields perfect classification on the training set, it is well known to produce highly irregular decision boundaries that are overly sensitive to noise and outliers in the data, leading to overfitting and poor generalization to unseen samples. By considering a neighborhood of five instances, the classifier is able to smooth the decision boundaries and reduce the influence of individual mislabeled or atypical points. This choice improves robustness and stability of the results while still preserving locality in the decision process. The use of a small odd value of k

Table 3. Average balanced accuracies for k-NN, SVM, XGBoost and Graph-NCC for 30 rounds of holdout 60%-40% with random sampling.

Dataset	1-NN	5-NN	SVM	XGBoost	Graph-NCC
Olivetti_Faces	0.8870 ± 0.0247	0.7283 ± 0.0331	0.0687 ± 0.0561	0.7345 ± 0.0387	0.8336 ± 0.0261
Satellite	0.8078 ± 0.0631	0.7316 ± 0.0581	0.500 ± 0.0000	0.7924 ± 0.0593	0.8016 ± 0.0535
one-hundred-plants	0.7277 ± 0.0120	0.7335 ± 0.0146	0.0205 ± 0.0108	0.6633 ± 0.0185	0.7610 ± 0.0150

such as $k = 5$, also avoids frequent ties in binary classification scenarios. Empirical studies and prior literature consistently suggest that moderate values of k (as 3, 5, or 7) provide a good balance between bias and variance.

A major strength of the proposed method lies in its ability to preserve the intrinsic geometry of the data manifold. By replacing Euclidean distances with geodesic approximations computed via shortest paths on a graph, Graph-NCC naturally adapts to nonlinear structures frequently encountered in real-world image datasets. The results highlight the importance of incorporating global structural information into local classification decisions.

Another notable advantage is the simplicity and interpretability of the method. Despite introducing a graph-based geodesic structure, the classifier maintains a computational profile similar to classical centroid-based methods, with the additional cost concentrated on graph construction and shortest-path computation. Furthermore, by adopting a prototype-based approach, Graph-NCC remains interpretable and suitable for scenarios where model transparency is valued.

However, the method has limitations. Its performance is sensitive to the graph construction process, particularly to parameter choices such as the number of nearest neighbors k in the k-NN graph and the scaling factor for edge weights. Suboptimal parameter choices may result in disconnected or overly dense graphs that fail to capture the true manifold structure. Additionally, scalability to large datasets can become an issue due to the cost of computing shortest paths between all points and prototypes. While the method is reasonably efficient for moderately sized datasets, further work is needed to extend it to large-scale problems, potentially through approximate nearest-neighbor strategies or landmark-based approaches. The Python code to run the experiments is available at (<https://github.com/alexandrelevada/Graph-NCC>).

4. Conclusions

Supervised image classification is a fundamental task in machine learning and computer vision. It involves training a model on a labeled dataset, learning the relationships between input features and their corresponding output labels, and subsequently making predictions on previously unseen instances. In this context, graph-based learning extends traditional approaches by incorporating information about the connectivity and proximity of data points through graphical representations. This integration allows models to exploit contextual information, uncover hidden patterns, and improve decision-making

by considering relationships among instances. By leveraging the inherent connections within the data, graph-based classification methods provide a unique perspective that can enhance both the accuracy and robustness of classification.

In this work, we presented a novel approach to image classification by developing a graph-based nearest centroid classifier (Graph-NCC). The method employs a k-NN graph to compute geodesic distances as a means to improve the performance of centroid-based classifiers, particularly in scenarios characterized by high-dimensional data.

Experimental results demonstrated the effectiveness of Graph-NCC across several real-world image datasets. The classifier exhibited a superior ability to learn from relatively small-sample problems, highlighting its potential for practical applications. The incorporation of geodesic distances enabled the model to capture the intrinsic structure of the data and improve classification accuracy. Compared to existing Euclidean distance-based classifiers such as k-NN, SVM, and XGBoost, Graph-NCC showed comparable performance, with slight superiority in some cases. The combination of graph-based representations and geodesic distances provides a flexible framework that can be extended to various machine learning domains, including medical image analysis, natural language processing, and bioinformatics.

The interpretability of Graph-NCC was also emphasized, as geodesic distances offer valuable insights into the relationships among data points, aiding in understanding the classification decisions. Additionally, the scalability and efficiency of the method were analyzed, demonstrating its applicability to large datasets and computationally demanding tasks.

This work bridges the gap between traditional centroid-based classifiers and modern machine learning. While NCC assumes linear separability and deep learning requires large n , Graph-NCC leverages graph-based geodesics to efficiently handle nonlinear high-dimensional data. This aligns with the manifold hypothesis while providing interpretability, a significant advantage over black-box methods such as XGBoost.

For future work, exploring Graph-NCC in semi-supervised and unsupervised learning settings is direction. Investigating the impact of different graph construction methods and geodesic distance metrics may provide additional opportunities for improvement. Comparisons with other graph-based classifiers, such as the Optimum Path Forest (OPF), also represent an interesting avenue of research. Finally, integrating Graph-NCC with combinatorial optimization algorithms for label propagation could pave the way for more sophisticated and expressive models in machine learning and data science.

Acknowledgements

Alexandre L. M. Levada thanks CNPq for the financial support through grant number 301432/2025-2. This work was partially funded by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) - Código de Financiamento 001.

Author Contributions

Single author.

References

- [1] DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern Classification*. 2. ed. New York: Wiley, 2001.
- [2] THEODORIDIS, S.; KOUTROUMBAS, K. *Pattern Recognition*. London: Academic Press, 2009.
- [3] JAMES, G. et al. *An Introduction to Statistical Learning with Applications in Python*. Cham: Springer, 2023. (Springer Texts in Statistics).
- [4] WEBB, A. R.; COPSEY, K. D. *Statistical Pattern Recognition*. London: John Wiley & Sons, 2011.
- [5] BRAGA-NETO, U. *Fundamentals of Pattern Recognition and Machine Learning*. Switzerland: Springer, 2020.
- [6] NARAYANAN, U. et al. A survey on various supervised classification algorithms. In: *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*. [S.l.: s.n.], 2017. p. 2118–2124.
- [7] SEN, P. C.; HAJRA, M.; GHOSH, M. Supervised classification algorithms in machine learning: A survey and review. In: MANDAL, J. K.; BHATTACHARYA, D. (Ed.). *Emerging Technology in Modelling and Graphics*. Singapore: Springer Singapore, 2020. p. 99–111.
- [8] MRABET, M. A. E.; MAKKAOUI, K. E.; FAIZE, A. Supervised machine learning: A survey. In: *2021 4th International Conference on Advanced Communication Technologies and Networking (CommNet)*. [S.l.: s.n.], 2021. p. 1–10.
- [9] TIBSHIRANI, R. et al. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academy of Sciences*, v. 99, n. 10, p. 6567–6572, 2002.
- [10] BAWONO, A. H.; BAHTIAR, F. A.; SUPIANTO, A. A. Nearest centroid classifier with outlier removal for classification. *Journal of Information Technology and Computer Science*, v. 5, n. 1, p. 57–64, Feb. 2020.
- [11] BEYER, K. et al. When is nearest neighbor meaningful? In: *Proceedings of the 7th International Conference on Database Theory*. Jerusalem, Israel: Springer, 1999. (ICDT '99), p. 217–235.
- [12] AGGARWAL, C. C.; HINNEBURG, A.; KEIM, D. A. On the surprising behavior of distance metrics in high dimensional space. In: *Lecture Notes in Computer Science*. Berlin, Heidelberg: Springer, 2001. v. 1973, p. 420–434.
- [13] CORMEN, T. H. et al. *Introduction to Algorithms*. 4th. ed. New York, NY, USA: The MIT Press, 2022. ISBN 9780262046305.
- [14] GORBAN, A. N.; TYUKIN, I. Y. Blessing of dimensionality: Mathematical foundations of the statistical physics of data. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Royal Society, v. 376, n. 2118, p. 20170237, 2018.
- [15] FEFERMAN, C.; MITTER, S.; NARAYANAN, H. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, American Mathematical Society, v. 29, n. 4, p. 983–1049, 2016.
- [16] SILVA, V.; TENENBAUM, J. Global versus local methods in nonlinear dimensionality reduction. In: BECKER, S.; THRUN, S.; OBERMAYER, K. (Ed.). *Advances in Neural Information Processing Systems*. Vancouver, Canada: MIT Press, 2002. v. 15.
- [17] SILVA, V. de; TENENBAUM, J. B. Global versus local methods in nonlinear dimensionality reduction. In: BECKER, S.; THRUN, S.; OBERMAYER, K. (Ed.). *Advances in Neural Information Processing Systems 15*. Cambridge, MA, USA: MIT Press, 2003. p. 721–728. Proceedings of the 16th International Conference on Neural Information Processing Systems (NIPS 2002). Disponível em: <https://papers.nips.cc/paper/2002/hash/6150ccc62edbce3142a5da63cc42e5e7-Abstract.html>.
- [18] PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Disponível em: <http://jmlr.org/papers/v12/pedregosa11a.html>.
- [19] ALPAYDIN, E.; ALIMOGLU, F. *Pen-Based Recognition of Handwritten Digits*. 1996. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C5MG6K>.
- [20] JAIN, A. K.; DUIN, R. P. W.; MAO, J. Statistical pattern recognition: A review. *IEEE Computer Society*, USA, v. 22, n. 1, p. 4–37, 2000.
- [21] ALPAYDIN, E.; KAYNAK, C. Cascading classifiers. *Kybernetika*, Institute of Information Theory and Automation AS CR, v. 34, n. 4, p. [369]–374, 1998.
- [22] SRINIVASAN, A. *Statlog (Landsat Satellite)*. 1993. UCI Machine Learning Repository. DOI: <https://doi.org/10.24432/C55887>.
- [23] BRODATZ, P. *Textures: A Photographic Album for Artists and Designers*. 1966. Dover, New York.
- [24] LINSSEN, M. K. L. (Ed.). *Plant Leaf Classification using Probabilistic Integration of Shape, Texture and Margin Features*, v. 798 de 098, (098, v. 798). [S.l.]: ACTA Press, 2013.
- [25] THE ORL Database of Faces. 2002. ATT Laboratories, Cambridge. Disponível em: <https://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>.