Integration of Manifold Learning and Density Estimation for Fine-Tuned Face Recognition

Huilin Ge 1, Zhiyu Zhu 1, Jiali Ouyang 1*, Muhammad Awais Ashraf 2, Zhiwen Qiu 1 and Umar Muhammad Ibrahim 3

1 School of Automation, Jiangsu University of Science and Technology, Zhenjiang 212003, China
2 School of Information Engineering, Chang’an University, Xi’an 710000, China
3 School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China

* Correspondence: oyjl1999@stu.just.edu.cn

Abstract: With the rapid advancements in data analysis and the increasing complexity of high-dimensional datasets, traditional dimensionality reduction techniques like Local Linear Embedding (LLE) often face challenges in maintaining accuracy and efficiency. This research aims to overcome the limitations of LLE, specifically its reliance on the nearest neighbor concept, its inability to distinguish differences among manifold points, and its underutilization of data discrimination information. To address these issues, we propose an advanced LLE algorithm that integrates decision tree-based neighbor recognition with Gaussian kernel density estimation. Decision trees accurately determine neighboring relationships, which are then optimized using Gaussian kernel density estimation to better reflect the distribution of sample points on the manifold. The algorithm also incorporates data discrimination information to enhance classification accuracy and efficiency. Evaluations in facial recognition tasks using SVM classifiers demonstrate significant improvements. Integrating decision trees (LLE-DT) yielded accuracy gains, with LFW at 98.75%, CFP 96.10%, and Olivetti 92.18%. Gaussian density estimation (LLE-GDE) achieved further enhancements, especially in LFW (99.13%), with CFP at 96.85%, and Olivetti at 91.82%. Combining both methods (LLE-DT-GDE) led to substantial improvements: LFW 99.61%, CFP 97.23%, and Olivetti 93.56%, highlighting the synergy between decision trees and Gaussian estimation. This advanced LLE algorithm effectively addresses the limitations of traditional approaches, showing promising results in complex data processing tasks such as facial recognition. These findings suggest its potential for broader applications in fields requiring robust data analysis and classification.

Keywords: manifold learning; high-dimensional data; facial recognition

1. Introduction

The integration of manifold learning, density estimation, and sparse representation for face recognition is engineered to enhance both the robustness and precision of the identification process. By utilizing sparse representation, the system is tailored to emphasize critical features that are resilient to environmental changes and obstructions, while manifold learning elucidates the intrinsic geometric structure of facial data, improving adaptability to various facial expressions and orientations. This amalgamation aims to significantly mitigate the challenges posed by variations in lighting, pose, expression, and occlusion, ensuring a robust framework capable of accurate face recognition under diverse conditions. Density estimation further refines this process by precisely determining the class membership of test samples based on their distribution within the feature space, thereby improving classification accuracy.
Moreover, this approach addresses the computational challenges associated with high-dimensional face data by reducing their dimensionality, thus simplifying the computational process and enhancing the system’s efficiency. The adaptive feature selection mechanism ensures the system focuses on the most relevant features, making the recognition process not only discriminative but also efficient. The integrated method’s capability to distinguish between known and unknown faces, coupled with its scalability and generalization across different datasets and conditions, marks a significant advancement in face recognition technology. It promises a scalable, reliable, and versatile solution that adapts seamlessly to a wide range of applications, setting a new benchmark for accuracy and adaptability in the field of face recognition.

Amidst the big data era, the challenges brought by high-dimensional data in areas like machine learning, data compression, pattern recognition, and data visualization have emerged as key areas of focus [1–3]. Especially in advanced pattern recognition domains like facial recognition, the escalation in data dimensions leads to an exponential increase in the needed sample size for efficacious statistical learning and feature identification [4,5]. As a result, the challenge of effectively learning and processing with limited high-dimensional samples (such as facial images with various angles and expressions) for more precise facial classification, feature denoising, and identity tracking has become a focal point of research.

Manifold learning algorithms constitute a set of effective nonlinear dimensionality reduction techniques [6,7]. These algorithms usually include building a neighborhood graph of data samples, a critical step in their execution [8,9]. In the conventional approach to constructing neighborhood graphs, methods such as k-nearest neighbors and ε-nearest neighbors are frequently used [10–12]. However, a significant drawback of these methods is their dependence on a predefined fixed number of neighbors, uniformly applied across all samples without taking into account the distinct neighborhood distribution features of each sample [13,14]. The concept of adaptive neighborhoods was introduced to address this limitation [15]. It centers around dynamically determining the size of the neighborhood based on the data distribution around the samples. In applications like face recognition within unsupervised learning, typical approaches encompass dimensionality reduction and density estimation. Techniques for dimensionality reduction, including Principal Component Analysis [16,17], Kernel-based PCA [18], and Multidimensional Scaling [19], effectively transform high-dimensional facial data into lower-dimensional formats, reducing computational complexity and preserving the essential information of facial features [20–22]. Furthermore, density estimation methods, be they parametric or non-parametric, aim to uncover the inherent distribution of facial data, facilitating the process.

Within parametric density estimation, although these approaches are significantly effective in approximating straightforward density functions [23,24], their precision might be constrained when handling complex features like multimodal distributions in facial characteristics [25,26]. In contrast, non-parametric density estimation methods demonstrate increased adaptability in mimicking the complex distributions of facial data [27]. Nonetheless, these non-parametric kernel estimation methods encounter the so-called “curse of dimensionality” when applied to high-dimensional facial data [28].

More specifically, the requisite sample size for density estimation in high-dimensional spaces (like multidimensional facial feature spaces) grows substantially with increasing dimensions [29–31], challenging the efficiency and precision of facial recognition technologies.

In this research, the identification of data points’ neighbors is based on decision trees, subsequently optimized by Gaussian kernel density estimation. Initially, the method utilizes decision trees to analyze the features of samples to recognize each sample’s local neighbors [32]. Then, Gaussian kernel density estimation is applied to refine the selection of neighbors, aiding in more precisely capturing the local structure and distribution char-
acteristics of the data. The integration of decision trees and Gaussian kernels not only elevates the precision in selecting neighbors but also boosts the classification capabilities of samples in low-dimensional spaces, leading to improved category segregation.

This method demonstrates significant advantages over traditional approaches due to several key factors. 1. Precision in neighbor selection is greatly enhanced by employing decision trees to analyze sample features and recognize local neighbors. This results in higher accuracy in identifying relevant data points, addressing the limitations of traditional nearest neighbor methods which often overlook subtle but significant relationships within the data. 2. The method further refines neighbor selection through the application of Gaussian kernel density estimation. This optimization step ensures a more accurate capture of the local structure and distribution characteristics of the data, a critical improvement over methods that rely solely on nearest neighbor concepts. 3. The integration of decision trees and Gaussian kernels not only enhances the accuracy of neighbor selection but also significantly boosts the overall classification capabilities of the algorithm. This dual approach effectively segregates categories in low-dimensional spaces, leading to improved classification performance. 4. This method effectively incorporates data discrimination information, unlike traditional LLE methods that do not fully leverage such information. This results in a more nuanced and efficient classification process, further enhancing the method’s robustness. 5. The ability of this method to precisely capture local structures and optimize neighbor relationships makes it particularly suitable for handling complex and high-dimensional datasets. Traditional methods often fall short in these scenarios, whereas this approach excels.

Overall, the combination of decision tree-based neighbor recognition and Gaussian kernel density estimation provides a robust framework that addresses the inherent limitations of other dimensionality reduction and classification methods. This method offers superior performance in terms of precision, efficiency, and effectiveness in data analysis tasks, making it a powerful tool for complex data processing.

2. Related Work

In the field of machine learning, samples within high-dimensional spaces are inclined towards sparsity, resulting in overfitting and the dimensionality dilemma. In high-dimensional feature spaces, most samples are situated at the edges of hypercubes, complicating classification with hyperplanes. A clear method to determine an “overly large” dimension is lacking. Approaches to address this include augmenting the sample size and diminishing the feature dimensions. The reduction in feature dimensions can be realized via feature selection and extraction, the latter of which develops new features encapsulating fundamental information, aiding in classifier simplification and improved generalizability to new data.

The processing of voluminous high-dimensional data is simplified by employing dimension reduction techniques. While linear dimension reduction maintains global features, it has limitations in handling complex data. Manifold learning reinstates low-dimensional structures, uncovering the inherent laws of data. Data often transition from low to high dimensions, akin to a two-dimensional sheet being rolled into a three-dimensional form [33]. Principal Component Analysis (PCA) [34] and Singular Value Decomposition (SVD) [35] are two commonly employed linear dimension reduction methods in dimensionality reduction strategies. PCA converts data into linearly independent variables, known as principal components, via orthogonal transformation, and SVD diminishes the dimensionality of data by decomposing the original data matrix. These approaches simultaneously decrease data volume and maintain the maximum variance in the data. Linear Discriminant Analysis (LDA) [36] is an additional prevalent linear dimension reduction method, focusing on identifying the optimal projection direction for maximum differentiation between various data categories. LDA is especially appropriate for classification tasks in the realm of supervised learning. Multidimensional Scaling (MDS) [37] aims to preserve the relative distances among samples from the original high-dimensional space.
in a reduced-dimensional setting, thus enabling MDS to maintain the original data’s structure to the greatest extent during dimensionality reduction. Locally Linear Embedding (LLE) [38] and Laplacian Eigenmaps (LE) [39] are two types of dimension reduction techniques grounded in manifold learning.

Considering these dimensionality reduction methods collectively, it becomes evident that each possesses unique strengths and suitable contexts for use. PCA and SVD are appropriate for global linear reduction of data dimensions, LDA is superior for dimensionality reduction in supervised classification problems, MDS emphasizes preserving the relative distances among samples during the reduction process, and LLE and LE are better suited for revealing the local geometric and topological structures of data. Hence, in practical applications, the choice of a suitable dimensionality reduction algorithm depends on the data’s attributes and the specific needs at hand. For example, LLE and LE may be more fitting in cases where retaining local characteristics is crucial, while PCA and MDS could be preferable when the maintenance of the data’s global structure is essential. Proper utilization of these methodologies is crucial for the effective handling and analysis of high-dimensional data.

Face recognition involves recognizing and analyzing local facial features, including the eyes, nose, and mouth. LLE is especially apt for maintaining the geometric and topological structures of such local features, as it retains the linear relationships between each data point and its nearest neighbors throughout the dimension reduction process. In contrast to global dimension reduction methods such as PCA and MDS, LLE is more adept at handling data with intricate local structures, as it more effectively captures nonlinear characteristics. Furthermore, facial recognition demands extensive computational resources to process high-dimensional image data. Utilizing LLE effectively reduces data dimensionality, thereby alleviating computational load while retaining crucial identification information. Nonetheless, LLE has its limitations, as the selection of the number of nearest neighbors greatly influences the outcomes of the LLE algorithm’s dimension reduction, but finding the ideal number of nearest neighbors is challenging and necessitates trial and error. Consequently, this study utilizes decision trees to adaptively select nearest neighbors and applies Gaussian density estimation for data optimization, introducing a modified version of the LLE algorithm.

Sparse representation stands as a complementary approach to the aforementioned dimensionality reduction techniques, offering a unique perspective on handling high-dimensional data, particularly in the context of face recognition. This method posits that high-dimensional data can often be represented as a sparse linear combination of basis elements from a dictionary, effectively reducing the complexity of the data while preserving essential information. Unlike traditional dimensionality reduction methods that focus on linear transformations or manifold learning techniques emphasizing the preservation of local or global data structures, sparse representation targets the sparsity principle to achieve data compression and feature selection. By identifying a minimal set of dictionary elements that can accurately represent each data sample, sparse representation facilitates robust classification and recognition by enhancing the discriminative power of the features.

In the domain of face recognition, sparse representation has demonstrated exceptional prowess in dealing with variations in illumination, pose, and expression, as well as occlusion. The technique’s ability to selectively focus on the most relevant features for representing a face enables the construction of compact yet highly informative feature vectors. This selective focus not only mitigates the curse of dimensionality but also improves the efficiency and effectiveness of face recognition systems. Furthermore, the adaptability of sparse representation to different data and noise levels makes it a versatile tool in the arsenal of dimensionality reduction and feature extraction methods. By integrating sparse representation with other dimensionality reduction strategies such as manifold learning, researchers can leverage the strengths of both approaches to develop more
sophisticated and capable face recognition systems that are both computationally efficient and highly accurate.

3. Theoretical Framework

3.1. Manifold Learning

Manifold learning in face recognition leverages the concept that faces, represented as high-dimensional data points (each pixel as a dimension), lie on a lower-dimensional manifold within this high-dimensional space. This approach is based on the observation that while images of faces can vary widely due to different lighting conditions, facial expressions, and angles, the intrinsic structure that defines a “face” is comparatively simple and can be captured in fewer dimensions. Manifold learning techniques reduce the dimensionality of face data to this intrinsic space, improving the efficiency and performance of face recognition algorithms by focusing on the most meaningful features.

Given a dataset \( X = \{x_1, x_2, \ldots, x_n\} \) in a high-dimensional space \( \mathbb{R}^d \) dimensionality reduction can be accomplished using an enhanced LLE algorithm. In the conventional k-nearest neighbors approach, the similarity between samples is determined based on their Euclidean distance, with each sample having an equal number of neighbors. The expression for the similarity coefficient is as follows:

\[
S_\beta(x, \text{ and } x_i \text{ are neighbors of each other}) \begin{cases} 
\alpha_{ij} = \exp(-\|x - x_j\|^2/\beta), & \|x - x_j\|^2 \text{ refers to the Euclidean distance between samples and } \beta \text{ denotes the squared mean of the Euclidean distances among all samples.} \\
0 (\text{others})
\end{cases}
\]

where \( \alpha_{ij} = \exp(-||x_i - x_j||^2/\beta) \). \( ||x_i - x_j||^2 \) refers to the Euclidean distance between samples and \( \beta \) denotes the squared mean of the Euclidean distances among all samples.

The k-nearest neighbors (k-NN) algorithm has three main limitations: Equal Importance to All Features: k-NN treats each feature with equal importance, not distinguishing which features are more crucial for classification or regression tasks. This can lead the algorithm to be influenced by irrelevant features, thereby reducing its performance. Distance-Based Neighbor Selection: The algorithm determines the nearest neighbors based on distance metrics, typically assuming that neighboring points in the data space have similar properties. However, this approach might not consider the nonlinear relationships between features, resulting in poor performance in identifying nonlinear boundaries in complex data structures. Interpretability Challenges: While the principle behind k-NN is simple and intuitive, its decision-making process can be less interpretable, especially in practical problems. This issue becomes more pronounced in high-dimensional spaces, where understanding why certain points are considered neighbors can be complex and non-intuitive.

3.1.1. Decision Tree-Based Nearest Neighbor Optimization

Decision trees are commonly employed for classification or regression issues, rather than for directly identifying the nearest neighbors of data points. Nonetheless, the outcomes of decision tree classifications can indirectly indicate the proximity relationships among data points. In the improved Local Linear Embedding (LLE) algorithm, decision trees help identify neighboring data points within the feature space, thus aiding in more efficient nonlinear dimensionality reduction.

Although decision trees are not traditionally used to determine nearest neighbors, we can indirectly use the results of decision tree classification to infer the neighbors of data points. The specific steps of this method are as follows:

(I) Develop a decision tree from the training dataset \( D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \) with \( x_i \) as the feature vector and \( y_i \) as the target variable. The decision tree determines features for division by maximizing Information Gain (IG) or minimizing Gini Impurity. The equation for Information Gain (IG) is as follows:
\[ IG(D_p, f) = I(D_p) - \frac{1}{N_p} \sum_{i=1}^{N_p} I(D_i) \]  

(2) In this context, \( D_p \) represents the dataset of the parent node, \( f \) denotes a particular feature, \( D_j \) is the dataset of the \( j \)-th child node created after splitting based on feature \( f \), \( N_p \) and \( N_j \) are the number of samples in the parent node and the \( j \)-th child node, respectively, and \( I \) is the function measuring impurity.

(3) Every leaf node symbolizes a set of samples from the training data, which are alike in the feature space. The identity of a leaf node can be determined by the split decisions along the path from the root to the leaf node. For a new query point \( x_q \), the decision tree is used to assign it to a specific leaf node. All training samples that fall into the same leaf node as \( x_q \) are considered to be the neighbors of \( x_q \).

3.1.2. Kernel Density Estimation for Enhanced Optimization

In the process of constructing neighbors for samples, those densely distributed in space demonstrate similar local features. For a given sample \( x_i \), a higher probability density of other samples in its vicinity indicates a larger number of samples with local features akin to \( x_i \), leading to a higher count of neighbors for \( x_i \). Consequently, this paper suggests the use of Gaussian Kernel Density Estimation (GKDE) for building a neighborhood graph and further refining and adjusting adaptive neighbor results based on the mean similarity coefficient \( M_i \). As a non-parametric probability density estimation technique, Gaussian Kernel Density Estimation can deduce the overall distribution probability density from the samples’ own data, especially when prior knowledge of sample distribution is lacking. In the high-dimensional space \( \mathbb{R}^D \), the Gaussian kernel estimation of neighborhood probability density for the sample \( x_i \) is expressed as:

\[ p(x_i) = \frac{1}{k^D} \frac{1}{(\sqrt{2\pi})^D} \exp\left(-\frac{d_i^2}{2k^2}\right) \]  

(3)

From Equation (3), the neighborhood probability density estimates \( p(x_i) \) for all samples are derived, leading to the calculation of the average neighborhood probability density

\[ \bar{p} = \frac{1}{k} \sum_{i=1}^{k} p(x_i) \]  

for all samples in the dataset \( X \). The adjustment of the neighbor count for the sample \( x_i \), based on the probability of neighborhood distribution, proceeds as follows:

\[ k_i^* = \text{floor}(k, p(x_i, \sqrt{p})) \]  

(4)

where ‘floor’ denotes rounding the data towards negative infinity.

According to Equation (4), it is evident that when the density of data samples around \( x_i \) is high, the count of its neighbors is automatically increased, whereas when the data distribution nearby is sparse, the neighbor count is automatically reduced.

Assuming that there is a low-dimensional manifold in the data, the density is estimated by learning the local structure on the manifold through the calculation of local covariance. If the neighboring points of each sample can reflect the local manifold features of that sample, thereby improving the density estimation, the function for density estimation is as follows:

\[ \hat{p}(x) = \frac{1}{l} N_{k,*} c(x) \]  

(5)
In this equation, \( N_{x_{i}^{p}}, c_{i}^{p}(x) \) is a multidimensional Gaussian kernel function, and \( c_{i}^{p} \) denotes the covariance matrix. The multidimensional Gaussian density function defined in Equation (5) is as follows:

\[
N_{x_{i}^{p}} c_{i}^{p}(x) = \frac{1}{\sqrt{(2\pi)^p \| C_{i} \|}} e^{-\frac{1}{2}(x-x_{i})^{T}C_{i}^{-1}(x-x_{i})}
\] (6)

In this equation, \( \| C_{i} \| \) is the rank of the covariance matrix \( C_{i} \). For a given sample point \( x_i \) its local geometric structure is disclosed through the distribution of \( x_i \) and its nearest neighbors. For each sample point \( x_i \), a local covariance matrix \( C_{i} \) is calculated by finding its k-nearest neighbors. In reality, if \( x_i \) and its surrounding points are densely distributed along a specific direction, the covariance values approach zero in orthogonal directions.

3.1.3. Improved LLE Method

The Locally Linear Embedding (LLE) algorithm is a well-known non-linear feature extraction method. Its core objective is to explore the critical structures within high-dimensional data and efficiently transfer these structures to a lower-dimensional space, emphasizing the essential properties of the data. Let \( X=[X_1, X_2, \ldots, X_N] \in \mathbb{R}^{D\times N} \) be the sample set in the high-dimensional space, where \( X_i=[x_{i1}, x_{i2}, \ldots, x_{iD}] \in \mathbb{R}^{D} \) denotes the i-th sample in the sample set, and \( i = 1, 2, \ldots, k, \ldots, N, D \) represent the feature dimensions of the high-dimensional samples, with N being the total number of samples. The detailed calculation steps are as follows:

1. In the process of improving the Local Linear Embedding (LLE) algorithm through the integration of decision trees, we first employ decision trees, rather than the conventional Euclidean distance, to ascertain the nearest neighbors for each sample point. This method provides a more thorough consideration of the multidimensional features of the samples, not just their spatial positioning. Decision trees intelligently choose neighbors with similar characteristics for each point by analyzing their attributes. This approach allows for more precise identification of each sample’s local linear areas, even within complex datasets, thereby enhancing the efficacy of the LLE algorithm in feature extraction.

2. Unearthing the local structure in the original data space. For instance, considering the sample point \( x_i \), for example, when examining the sample point \( x_i \), its high-dimensional local reconstruction model can be formally represented as follows:

This is example 1 of an equation:

\[
\min \frac{1}{2} \| x_i - \sum_{j=1}^{k} W_{ij} x_j^{'} \|_2^2 = \frac{1}{2} \| x_i - x_i^{'} w_i \|_2^2
\]

s.t \( e' W_i = 1 \) (7)

In this context, \( e = [1, 1, \ldots, 1] \in \mathbb{R}^{D} \), \( x_i \) refers to the j-th nearest neighbor of sample point \( x_i \), \( x_i^{'}=[x_i^{'}_1, x_i^{'}_2, \ldots, x_i^{'}_D] \) represents the set of nearest neighbors for point \( x_i \), and \( W_i \) is the reconstruction weight between point \( x_i \) and its j-th nearest neighbor \( x_i \). Imposing constraints can stabilize the results. Thus, models with constraints can be solved using the method of Lagrangian multipliers. The model of Lagrangian multipliers is defined as follows:

\[
L(W_i, \lambda_i) = \frac{1}{2} \| x_i - x_i^{'} w_i \|_2^2 + \lambda_i (e' W_i - 1)
\]

(8)

The method to compute the aforementioned equation involves taking the partial derivatives of the parameters on the left side of the equation and setting the results to zero:
\[ x' (x_i - x_j^W) + \lambda_i e = 0 \]  \hspace{1cm} (9)

\[ e' W_i - 1 = 0 \]  \hspace{1cm} (10)

Here, \( \lambda_i \) represents the Lagrange multiplier. The optimal solution \( W_i \) for the objective function (7) is as follows:

\[ W_i = \frac{G_i^{-1} e}{e' G_i^{-1} e} \]  \hspace{1cm} (11)

where \( G_i = [x_i - x_i^1, x_i - x_i^2, \cdots, x_i - x_i^N] [x_i - x_i^1, x_i - x_i^2, \cdots, x_i - x_i^N] \in R^{N \times k} \) is calculated as the local covariance matrix.

(3) Compute the low-dimensional embedding results for the samples. This is achieved by maintaining the local structure of the high-dimensional data unchanged and performing linear reconstruction in a low-dimensional space, with the reconstruction error function being

\[

c_{i} \sum_{j=1}^{N} |Y_{j} - \sum_{i=1}^{N} w_{j i} Y_{i}|^{2} \\
s.t. \sum_{j=1}^{N} Y_{j} = 0, \frac{1}{N} \sum_{j=1}^{N} Y_{j} Y_{j} = I_{N} \]  \hspace{1cm} (12)

In this context, \( Y_{j} \) denotes the low-dimensional embedding result corresponding to \( X_{j} \). \( Y_{j} \) is the \( j \)-th nearest neighbor of \( Y_{i} \), \( j \in R^{k} \) signifies the set of labels \( (i, j = 1, 2, \ldots, N) \) for \( X_{i} \)’s nearest neighbors, and \( d \) is the dimension of the embedding result. Imposing two constraints on the objective function ensures the stability of the results. Solving the objective function (12) using related algorithms would increase computational complexity, thus it can be rephrased as follows:

\[

c_{i} \sum_{j=1}^{N} ||Y_{j} - \sum_{i=1}^{N} w_{j i} Y_{i}||^{2} = Y(I_{N} - W)(I_{N} - W)Y = YMY' \]  \hspace{1cm} (13)

where \( Y = [Y_{1}, Y_{2}, \cdots, Y_{N}] \), \( W \in R^{N \times N} \) are the local reconstruction weight matrices, with elements \( w_{ij} = (i, j = 1, 2, \ldots, N) \) in \( W \). If \( X_{i} \) is a neighboring point of \( X_{i} \), \( w_{ij} \) can be calculated using the above equation; if not, then \( w_{ij} = 0 \), \( i = 0 \), and \( M \) are large sparse symmetric matrices. Based on the relationship between eigenvalues and eigenvectors, it follows that

\[ M = diag[\lambda_{1}, \lambda_{2}, \cdots, \lambda_{k}] \]  \hspace{1cm} (14)

where \( P \) represents the eigenvectors corresponding to the eigenvalues calculated from matrix \( M \), \( \Lambda \) denotes the eigenvalues, and they are mutually orthogonal to each other, so multiplying by \( P' \) yields

\[ P'MP = diag[\lambda_{1}, \lambda_{2}, \cdots, \lambda_{k}] \]  \hspace{1cm} (15)

Thus, it becomes evident that the matrix composed of the eigenvectors associated with the eigenvalues derived from the objective function (12) serves as the transformation matrix, using which the ultimate embedding result is achieved.

3.1.4. Support Vector Machine

The Support Vector Machine (SVM), simply put, is a classification model. Its basic model is defined as the linear classifier with the maximum margin in the feature space. Its learning strategy is to maximize the margin, which ultimately can be transformed into solving a convex quadratic programming problem. The classical solution method is the Lagrange multiplier method, as shown in Equation (16):

\[ x' (x_i - x_j^W) + \lambda_i e = 0 \]  \hspace{1cm} (9)

\[ e' W_i - 1 = 0 \]  \hspace{1cm} (10)
\[ L(W,c,d) = \frac{1}{2} ||W||^2 - \sum_{i=1}^{N} c_i \{y_i (\langle X_i \cdot W \rangle + d) - 1 \} \]  

(16)

in which \( W \) is the coefficient vector and \( d \) is a constant.

Taking partial derivatives of \( W \) and \( d \) yields Equation (17):

\[ \max H(c) = \sum_{i=1}^{N} c_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j c_i c_j \langle X_i \cdot X_j \rangle \]  

(17)

Solving this equation yields the vector \( W^* \), as shown in Equation (18):

\[ W^* = \sum_{i=1}^{N} y_i c_i \cdot X_i \]  

(18)

where \( y_i \) represent the vertical coordinates of points \( i \) and \( j \), \( c_i \) represent the offsets of points \( i \) and \( j \), \( X_i \) and \( X_j \) are the image feature pixel points, and \( H(c) \) is the obtained discriminant matrix. Solve for the optimal values of \( c^* \) and \( d^* \), as well as the optimal discriminant function. The optimal \( c^* \) is determined by the constraint in Equation (19), and \( c^* \) and \( W^* \) can be obtained using a quadratic programming algorithm. Then, by selecting a support vector \( X \), the value of \( d^* \) can be found, as shown in Equation (20), resulting in the final optimal discriminant function as shown in Equation (21).

\[ c^* \{y_i (\langle W^* \cdot X_i \rangle + d^*) - 1\} = 0 \]  

(19)

\[ d^* = y_i - \langle W^* \cdot X_i \rangle \]  

(20)

\[ f(X) = \sum_{i=1}^{N} y_i c_i \cdot \langle X \cdot X_i \rangle + d^* \]  

(21)

where \( W^* \) is the matrix obtained after differentiation, \( c^* \) is the optimal offset, and \( d^* \) is the obtained optimal correction value.

To address the difficulty of finding a corresponding classification hyperplane in a low-dimensional plane for the dataset, this article introduces a kernel function to transform the data into a high-dimensional plane, which is advantageous for finding a more effective classification plane. The specific process is illustrated in the diagram below (Figure 1).

In summary, the entire process involves three key steps: starting with the collection of high-dimensional sample input, applying manifold learning algorithms for dimensionality reduction (as depicted in Figure 2), and finally using an SVM classifier for accurate classification. This combination of techniques leverages the strengths of manifold learning to simplify the data and the robustness of SVM for effective classification, thereby improving the overall performance of the system.
Figure 1. (a) Potentially occluded or (b) corrupted, as a sparse linear combination of all the training images (middle) plus sparse errors (right) due to occlusion or corruption. Green (darker) coefficients correspond to training images of the correct individual. The algorithm determines the true identity (indicated with a red box at second row and third column) from 700 training images of 100 individuals (7 each) in the standard AR face database.

Figure 2. Flow diagram of the manifold learning algorithm.

3.2. Sparse Representation Enhancement

Sparse representation refers to expressing a signal or data sample as a linear combination of a few essential elements or features from a given dictionary, where the majority of coefficients in the combination are zero. In other words, sparse representation aims to represent data efficiently by using only a small number of significant components from a larger set of possibilities. This approach is valuable in various fields, including signal processing, image analysis, and machine learning, as it helps in reducing data complexity, extracting essential information, and enhancing computational efficiency. The sparsest representation is naturally discriminative; among all subsets of base vectors, it selects the subset which most compactly expresses the input signal and rejects all other possible but less compact representations.

This approach assumes that if adequate training samples are available for each class, a test sample can be represented as a linear combination of only those training samples that belong to the same class, resulting in a naturally sparse representation. This sparsity involves only a small fraction of the training database and is posited to be the sparsest linear representation for the test sample within this dictionary, which can be efficiently identified through $\ell^1$-minimization.

The proposed classifier is considered a generalization of existing classifiers such as nearest neighbor (NN) and nearest subspace (NS), offering a more flexible approach that adaptively chooses the minimal number of training samples needed for representing each test sample. This method achieves a balance similar to the nearest feature line (NFL) algorithm but extends the concept by considering all possible combinations of training samples, both within and across classes, to achieve optimal sparse representation.
Further analysis involves evaluating the residuals, $r(y)$, for a test image of subject 1, which are calculated with respect to the projected sparse coefficients, $\hat{\theta}(\mathcal{X})$, obtained through $l^1$-minimization. The comparison of the two smallest residuals reveals a ratio of approximately 1:8.6, highlighting the precision of the classification method in distinguishing between subjects based on sparse representation techniques. This ratio underscores the method’s ability to accurately identify and differentiate between subjects with a high degree of confidence.

The algorithmic process for enhancing face detection through sparse representation is a sophisticated approach that leverages mathematical and computational techniques to improve the accuracy and robustness of identifying faces within digital images. Here is a detailed explanation of each step in the process:

3.2.1. Dictionary Creation

The foundation of sparse representation is the construction of a dictionary composed of basis elements. These elements are essentially a set of vectors that span the space in which facial images reside. The dictionary can be derived from training images, capturing the variability of facial features, or generated through methods like principal component analysis (PCA) or independent component analysis (ICA), which identify statistically significant bases. A crucial aspect of the dictionary is its overcompleteness; it contains more elements than the dimensionality of the input space, enabling a more flexible and expressive representation of facial images. Overcompleteness is key to achieving sparsity, as it allows each image to be represented by a small subset of the dictionary elements.

3.2.2. Preprocessing of Input Images

Before representation, input images undergo preprocessing to standardize their format and enhance feature visibility. This typically involves normalizing the intensity values to ensure consistency across the dataset, resizing images to a uniform dimension to simplify processing, and converting them to grayscale to reduce the computational load. Optionally, feature extraction techniques can be applied to distill essential information from the images, focusing on characteristics crucial for face detection, such as edges or texture patterns. This step reduces the dimensionality of the data, making the subsequent sparse representation more computationally efficient.

3.2.3. Sparse Coding of Facial Images

Sparse coding is the core of this process, where each facial image is represented as a sparse linear combination of the dictionary elements. This involves solving an optimization problem to find a coefficient vector that combines dictionary elements to reconstruct the input image with minimal error, while enforcing sparsity constraints on the coefficients. Techniques like Lasso, Orthogonal Matching Pursuit (OMP), or Basis Pursuit are employed to achieve this balance, ensuring that the resulting representation uses the fewest possible dictionary elements. This sparsity is crucial for highlighting the unique aspects of each face and facilitating efficient classification.

3.2.4. Face Classification/Recognition

Once an image is represented in sparse form, the resulting coefficient vector serves as a feature vector for classification or recognition tasks. This vector’s sparsity pattern—which elements are non-zero and their magnitudes—provides a powerful discriminative signature for each face. A classifier trained on these sparse representations learns to distinguish between different individuals based on these signatures. Various machine learning models, including support vector machines (SVM), k-nearest neighbors (k-NN), or neural networks, can be used depending on the specific requirements of accuracy, speed, and scalability.
3.2.5. Detection and Localization in New Images

For detecting faces within new, unseen images, a sliding window technique is often employed. This method involves moving a fixed-size window across the image, applying the sparse coding and classification process to each windowed segment to determine whether it contains a face. This approach allows for the localization of faces within larger images by identifying and classifying regions based on their sparse representation.

3.2.6. Post-Processing

After initial detection, post-processing steps such as non-maximum suppression are used to refine the results (Figure 3). This can involve eliminating redundant detections that overlap significantly, ensuring that each face is only marked once and that the bounding boxes accurately represent the location and extent of faces in the image. Adjusting the classification confidence threshold is another post-processing strategy, balancing the detection rate against the false positive rate to meet the specific needs of the application.

![Diagram](image)

**Figure 3.** Processing After Detection Refinement techniques like non-maximum suppression are used after first detection to improve the results.

Sparse representation in image recognition involves representing images as linear combinations of a few basis elements from a large dictionary. The core mathematical formulations related to sparse representation in this context are centered around solving optimization problems that seek the sparsest coefficients to reconstruct the input images from the dictionary. Below are the key formulas and concepts used in sparse representation for image recognition:

**Sparse Coding:**

Given a dictionary $D \in \mathbb{R}^{m \times n}$, where $m$ is the dimensionality of the feature space and $n$ is the number of dictionary elements (with $n > m$ typically, making $D$ overcomplete), and an image (or image feature vector) $y \in \mathbb{R}^m$, sparse coding aims to find the sparsest coefficient vector $x \in \mathbb{R}^n$ such that $y$ approximates $Dx$. The optimization problem can be formulated as follows:

$$ \hat{x} = \arg\min_x \|x\|_0 $$

subject to $\|Dx - y\|_2^2 \leq \epsilon$

where $\|x\|_0$ denotes the $l_0$ norm, which counts the number of non-zero entries in $x$ and $\epsilon$ is a tolerance level for the reconstruction error.
3.2.7. L1 Minimization

Since the $l_0$ minimization is NP-hard and computationally infeasible for large problems, a common approximation is to use the $l_1$ norm instead, leading to the basis pursuit problem.

$$\hat{x} = \arg\min_x \|x\|_1$$

subject to $\|Dx - y\|_2^2 \leq \epsilon$

This problem promotes sparsity in $x$ due to the properties of the $l_1$ norm and can be solved more efficiently.

**Lasso (Least Absolute Shrinkage and Selection Operator):**

Another popular formulation for finding sparse representations is the Lasso, which directly incorporates the error term into the objective function.

$$\hat{x} = \arg\min_x \frac{1}{2} \|Dx - y\|_2^2 + \lambda \|x\|_1$$

where lambda is a regularization parameter that controls the trade-off between the reconstruction error and the sparsity of $x$.

**Orthogonal Matching Pursuit (OMP):**

OMP is a greedy algorithm that iteratively selects the dictionary elements most correlated with the current residual (the difference between the input image and its current sparse approximation). At each step, it adjusts the coefficients of the selected elements to best fit the input image:

1. Initialize the residual $r^0 = y$ and the support set $S = \emptyset$.
2. At each step $k$, find the dictionary element $d_j$ most correlated with the residual $r^{k-1}$ and add $j$ to $S$.
3. Solve a least squares problem to update $x$ using only elements in $S$.
4. Update the residual $r^k = y - Dx$.
5. Repeat steps 2–4 until a stopping criterion is met (e.g., a desired number of elements has been selected or the residual is below a threshold).

These formulations and algorithms underlie the application of sparse representation in image recognition, providing a framework for representing images with a few significant features extracted from a comprehensive dictionary, thereby facilitating efficient and effective recognition.

4. Proposed Method

The proposed approach incorporates the integration of manifold learning and sparse representation which presents a sophisticated approach for enhancing image recognition tasks, particularly in the domain of face recognition. Manifold learning’s ability to unveil the intrinsic low-dimensional structures hidden within high-dimensional data complements the efficiency of sparse representation, which encodes images as sparse linear combinations of basis elements from a pre-defined dictionary. This synergy not only significantly reduces the computational complexity associated with processing and analyzing high-dimensional datasets but also improves the accuracy and robustness of recognition algorithms. By effectively capturing the essential features of images through manifold reduction and then succinctly representing these features in a sparse format, this integrated approach offers a powerful solution for identifying and classifying images with high precision, even in the presence of variability and noise. The process is shown in Figure 4.
Figure 4. Summary of the manifold learning and sparse representation proposed methodology.

**Step 1**: Preparing the Dataset

Collect a Diverse Dataset: gather a large and varied set of facial images that include different expressions, lighting conditions, and angles to ensure the robustness of your model.

Preprocess the Images: Normalize the images to have the same size and scale. Convert them to grayscale to reduce computational complexity and apply other preprocessing techniques as necessary to enhance image quality.

**Step 2**: Applying Manifold Learning for Dimensionality Reduction

Select a Manifold Learning Algorithm: choose an algorithm like Locally Linear Embedding (LLE), Isometric Mapping (Isomap), or Uniform Manifold Approximation and Projection (UMAP) based on the specific needs of your dataset and task.

Reduce the Dimensionality: Apply the chosen manifold learning algorithm to the dataset to reduce its dimensionality. This step aims to retain the essential structure of the data while projecting them onto a lower-dimensional space.

**Step 3**: Constructing the Dictionary for Sparse Representation

Dictionary Creation: Build a dictionary using a subset of the reduced-dimensional data. This dictionary can also be learned from the data using algorithms like K-SVD.

Overcompleteness: ensure the dictionary is overcomplete (has more atoms than the dimensionality of the feature space) to allow for a rich representation of faces.

**Step 4**: Sparse Coding of Faces

Encode Faces: For each reduced-dimension face image, find the sparsest representation by selecting the minimal number of dictionary atoms that combine to approximate the face. This involves solving an optimization problem, often using L1 minimization (Lasso) or Orthogonal Matching Pursuit (OMP).

Feature Vector: the sparse coefficient vector resulting from this encoding serves as the feature vector for each face, encapsulating its most distinguishing characteristics.

**Step 5**: Classification and Recognition

Training a Classifier: Use the sparse representations as features to train a classifier. Support Vector Machines (SVM) or Nearest Neighbor classifiers are common choices.

Face Recognition: For a new face image, repeat the preprocessing, dimensionality reduction, and sparse coding steps to obtain its sparse representation. Then use the trained classifier to determine the identity of the face based on its sparse feature vector.

**Step 6**: Evaluation and Refinement

Test the System: Evaluate the performance of your face recognition system using a separate test dataset not seen during training. Common metrics include accuracy, precision, recall, and F1 score.
Refine the Model: Based on the evaluation, refine the model by adjusting parameters, choosing different manifold learning algorithms, or modifying the dictionary. Iteratively improve the system for better performance.

5. Results

In order to thoroughly assess the effectiveness and practicality of the approach presented in this research, this experiment utilized three diverse datasets, a. Labeled Faces in the Wild (LFW) [40], b. Celebrities in Frontal-Profile (CFP) [41], and c. the Olivetti Faces Dataset [42], and performed comparative experiments with six varied algorithms. These datasets include multiple types and sizes to guarantee the extensive applicability and dependability of the experimental outcomes.

5.1. Improved LLE Method

The Labeled Faces in the Wild (LFW) dataset comprises over 13,000 facial images, encompassing over 5000 distinct individuals. These images represent a range of ages, ethnicities, and genders. The diversity of the images, featuring variations in illumination, facial expressions, poses, and obstructions, renders it an excellent choice for evaluating face recognition algorithms in complex settings, which motivated its selection for this research. Introduced by Sengupta and others in 2016, the CFP (Celebrities in Frontal-Profile) dataset [28] aims to verify the accuracy of face recognition algorithms in the transition from frontal to profile postures. It contains 500 identities, with each identity having 10 frontal and 4 profile images. Like the LFW dataset, it is segmented into 10 subsets, with each subset comprising 350 pairs of matching samples and 350 pairs of non-matching samples. Created by the Olivetti Research Laboratory, the Olivetti Faces Dataset features 400 grayscale facial images of 40 distinct individuals. Each individual is represented with 10 images, showcasing a variety of expressions and poses. A notable aspect of this dataset is the uniformity in posture. LFW, CFP, Olivetti samples are shown in Figure 5.

![Figure 5](image_url)

**Figure 5.** Sample images from LFW, CFP, and Olivetti datasets for improved LLE method.

5.2. Dimensionality Reduction Analysis

For a clearer demonstration of how the LLE Improvement Algorithm Based on Decision Trees and Gaussian Density Estimation (LLE-DT-GDE) method processes input images through feature extraction, visualization of the output feature maps from the PCA, SVD, LDA, MDS, LLE, LE, and LLE-DT-GDE methods was conducted. As depicted in Figure 6, the method proposed in this study captures and extracts finer detail information...
in comparison to other methods, thus contributing to improved facial expression recognition results [43]. The visualizations demonstrate that the method proposed, by employing both global enhancement and local attentive features, more effectively concentrates on distinguishable facial regions like the eyes, mouth, and eyebrows. This allows for the retention of crucial and supplementary information for recognition, consequently enhancing the algorithm’s capacity for accurate identification.

Figure 6. Illustrates a case involving an irrelevant test image. (a) This section displays the sparse coefficients obtained for the irrelevant test image when compared to the same training dataset used in Example 1. The test image was chosen at random and does not pertain to the dataset. (b) This part shows the residuals for the irrelevant test image relative to the projection $\delta_0(\mathbf{x})$ from the sparse representation determined through $l^1$-minimization. The smallest two residuals have a ratio of approximately 1:1.2.

To showcase the face recognition abilities of our approach, ten distinct identities were chosen from the LFW dataset, each including images with either no obstruction or synthetic mask obstructions. Feature extraction was performed on these images to obtain 512-dimensional features, subsequently reduced to 2 dimensions via PCA, SVD, LDA, MDS, LLE, LE, and the LLE-DT-GDE method. Figure 7 illustrates the recognition process utilizing image-derived features, demonstrating how these features are processed through the sparse coding and dictionary learning framework to achieve accurate recognition. The visualization of these features, depicted in Figure 8, underscores the efficacy of our method. The results show that PCA, SVD, LDA, and MDS have smaller inter-class gaps and less compact intra-class grouping, which are not ideal for facial recognition. On the other hand, LE, LLE, and the method proposed in this study demonstrate the opposite, especially the latter, which distinctly separates different identities with clear boundaries and accurately clusters various images of the same identity, thus validating the effectiveness of our method.
For an objective evaluation of the effectiveness of the method presented in this study, we used an SVM classifier and devised six contrasting methods. Comparisons were made between the LLE-DT-GDE+SVM method and PCA+SVM, SVD+SVM, LDA+SVM, MDS+SVM, LLE+SVM, and LE+SVM algorithms to validate the efficacy of our approach.

From Table 1, it is clear that the method introduced in this study records average accuracies of 99.61%, 97.23%, and 93.56% on the LFW, CFP, and Olivetti datasets, respectively, surpassing the performance of other methods. The enhanced LLE method includes two components: one is the use of decision trees to calculate nearest neighbor relations, replacing traditional approaches, and the other is the application of Gaussian density estimation for further optimization of these relations. Relative to the original LLE, there are accuracy improvements of 1.09, 0.68, and 3.57 percentage points, respectively. The results demonstrate that the refined LLE algorithm improves facial recognition rates and fosters intra-class compactness.

Table 1. Comparative performance of various algorithms on the LFW, CFP, and Olivetti face databases.

<table>
<thead>
<tr>
<th>Methods</th>
<th>LFW</th>
<th>CFP</th>
<th>Olivetti</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA+SVM</td>
<td>95.88%</td>
<td>92.45%</td>
<td>88.63%</td>
</tr>
<tr>
<td>SVD+SVM</td>
<td>94.22%</td>
<td>91.99%</td>
<td>87.12%</td>
</tr>
<tr>
<td>LDA+SVM</td>
<td>94.62%</td>
<td>92.03%</td>
<td>87.96%</td>
</tr>
<tr>
<td>MDS+SVM</td>
<td>95.83%</td>
<td>91.78%</td>
<td>88.52%</td>
</tr>
<tr>
<td>LLE+SVM</td>
<td>98.52%</td>
<td>96.55%</td>
<td>89.99%</td>
</tr>
<tr>
<td>LE+SVM</td>
<td>98.45%</td>
<td>96.88%</td>
<td>90.02%</td>
</tr>
<tr>
<td>Ours</td>
<td>99.61%</td>
<td>97.23%</td>
<td>93.56%</td>
</tr>
</tbody>
</table>
To further validate our approach, we compared the improved graph representations of Sparse Representation-based Classification (SRC), linear SVM, nearest neighbor, and nearest subspace methods, as shown in Figure 9. The results demonstrate that our method outperforms these traditional approaches, providing clearer and more distinct class separations, thus enhancing recognition accuracy.

![Figure 9. Improved graph representations of SRC, linear SVM, nearest neighbor, and nearest subspace.](image)

### 6. Comparison Study

To systematically assess the contributions of the suggested enhancements—decision tree integration and Gaussian kernel density estimation (GDE)—to the conventional Locally Linear Embedding (LLE) approach, we carried out comparison study in this section. Our goal was to identify the relative contributions of each improvement to the overall performance of the upgraded LLE algorithm.

#### 6.1. Decision Tree Integration

In order to isolate the effect of adding decision trees to the LLE algorithm, we ran tests where we only used decision trees to calculate the nearest neighbor—that is, we did not use Gaussian kernel density estimation. “LLE-DT” is the designation given to this algorithmic variation.

From Table 2, the error rates of LLE-DT on the LFW, CFP, and Olivetti datasets are 1.25%, 3.9%, and 7.82% respectively, which are notably lower compared to those achieved by the traditional LLE algorithm. The comparison demonstrates that decision trees are
effective in capturing nearest neighbor interactions and boosting the discriminative capability of the LLE algorithm. It also shows that adding decision trees alone results in a slight improvement in accuracy across all datasets.

### Table 2. Comparative performance of LLE-DT on the various face databases.

<table>
<thead>
<tr>
<th>Methods</th>
<th>LFW Accuracy</th>
<th>CFP Accuracy</th>
<th>Olivetti Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>LLE-DT</td>
<td>98.75%</td>
<td>96.10%</td>
<td>92.18%</td>
</tr>
<tr>
<td>Traditional LLE</td>
<td>98.52%</td>
<td>96.55%</td>
<td>89.99%</td>
</tr>
</tbody>
</table>

#### 6.2. Gaussian Density Estimation (GDE)

As shown in Table 3, we then investigated how Gaussian kernel density estimation (GDE) affects the LLE algorithm by using GDE only for density estimation in place of decision trees. This variation has the designation “LLE-GDE”.

### Table 3. Comparative performance of LLE-GDE on the various face databases.

<table>
<thead>
<tr>
<th>Methods</th>
<th>LFW Accuracy</th>
<th>CFP Accuracy</th>
<th>Olivetti Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>LLE-GDE</td>
<td>99.13%</td>
<td>96.85%</td>
<td>91.82%</td>
</tr>
<tr>
<td>Traditional LLE</td>
<td>98.52%</td>
<td>96.55%</td>
<td>89.99%</td>
</tr>
</tbody>
</table>

As shown in Figure 3, the findings show that Gaussian kernel density estimation makes a substantial contribution to the accuracy gain, which is especially noticeable in the LFW dataset. The error rate is merely 0.87%, demonstrating a significant improvement in accuracy. This emphasizes how crucial density estimation is to improving the proximity associations that the LLE algorithm detects.

#### 6.3. Combined Enhancement (LLE-DT-GDE)

As shown in Table 4, we assessed the overall enhancement attained through the combination of decision trees and Gaussian kernel density estimation within the LLE technique. The full suggested method is represented by this setup.

### Table 4. Comparative performance of LLE-DT-GDE on the various face databases.

<table>
<thead>
<tr>
<th>Methods</th>
<th>LFW Accuracy</th>
<th>CFP Accuracy</th>
<th>Olivetti Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>LLE-DT-GDE (Ours)</td>
<td>99.61%</td>
<td>97.23%</td>
<td>93.56%</td>
</tr>
<tr>
<td>Traditional LLE</td>
<td>98.52%</td>
<td>96.55%</td>
<td>89.99%</td>
</tr>
</tbody>
</table>

As shown in Figure 4, the comparison demonstrates how decision trees and Gaussian density estimation work well together to produce significant accuracy gains on all datasets. The effectiveness of the suggested improvements is confirmed by the suggested method’s persistent superiority over the conventional LLE algorithm.

#### 6.4. Discussion

In this section, we systematically assessed the contributions of integrating decision trees and Gaussian kernel density estimation (GDE) into the conventional Locally Linear Embedding (LLE) algorithm through an ablation study. Our primary goal was to identify the relative contributions of each enhancement to the overall performance of the upgraded LLE algorithm.

Our results indicate that the integration of decision trees into the LLE algorithm (referred to as LLE-DT) leads to a noticeable improvement in accuracy across all datasets tested. Specifically, LLE-DT achieved accuracies of 98.75% on the LFW dataset, 96.10% on the CFP dataset, and 92.18% on the Olivetti dataset, compared to the traditional LLE’s accuracies of 98.52%, 96.55%, and 89.99%, respectively. The decision trees effectively capture nearest neighbor interactions, which enhances the discriminative capability of the
LLE algorithm. These studies have demonstrated the utility of decision trees in improving classification tasks by providing robust mechanisms for nearest neighbor searches.

When examining the effect of Gaussian kernel density estimation (GDE) on the LLE algorithm (designated as LLE-GDE), we observed a substantial contribution to accuracy improvement. The LLE-GDE variation achieved accuracies of 99.13% on the LFW dataset, 96.85% on the CFP dataset, and 91.82% on the Olivetti dataset. This underscores the significance of effective density estimation in enhancing the proximity associations identified by the LLE algorithm. GDE’s ability to provide a smoother and more nuanced understanding of the data distribution contributes significantly to these accuracy gains, particularly in the LFW dataset, which is known for its challenging face recognition scenarios.

LLE-DT-GDE, integrating both decision trees and Gaussian kernel density estimation within the LLE technique (denoted as LLE-DT-GDE), demonstrated the most significant accuracy improvements across all datasets. This combined method achieved accuracies of 99.61% on the LFW dataset, 97.23% on the CFP dataset, and 93.56% on the Olivetti dataset, outperforming the traditional LLE algorithm, which achieved 98.52%, 96.55%, and 89.99%, respectively. The synergy between decision trees and GDE is evident, as they complement each other in capturing complex data structures and enhancing the overall performance of the LLE algorithm.

The ablation study confirms that both decision tree integration and Gaussian kernel density estimation individually contribute to improving the accuracy of the LLE algorithm. However, their combined application (LLE-DT-GDE) yields the most significant performance gains, demonstrating the potential of hybrid approaches in enhancing manifold learning techniques. These findings contribute to the broader understanding of how traditional algorithms can be effectively upgraded through strategic enhancements, offering valuable insights for future research and applications in machine learning and pattern recognition.

6.5. Future Directions

Subsequent investigations will concentrate on optimizing the suggested approach to improve computing effectiveness, especially when working with extensive datasets. Furthermore, investigating different density estimation methods and integrating with sophisticated machine learning frameworks can produce more performance improvements and broaden the algorithm’s application to a variety of real-world situations.

7. Future Directions and Conclusions

In this research, an advanced face recognition algorithm is proposed, which integrates decision tree and Gaussian kernel density estimation methods to more precisely capture the relationships between nearest data points, thus significantly enhancing the accuracy of facial recognition. More specifically, the algorithm initially employs the robust classification abilities of decision trees to accurately ascertain the nearest neighbor relationships among the data points. Decision trees, an effective machine learning approach, can swiftly and accurately classify and process data by developing a sequence of decision-making rules. Within this algorithm, the utilization of decision trees offers a distinct map of the proximity relationships among data points, a crucial step for comprehending and analyzing intricate data structures. The algorithm enhances these neighboring relationships through Gaussian kernel density estimation. Gaussian kernel density estimation is a non-parametric method for estimating probability density functions. It smooths and optimizes the distribution of sample points, resulting in a finer and more faithful representation of the distribution of sample points on the manifold. In the context of facial recognition, this step is especially crucial as it can unveil latent non-linear structural features within the data, which are vital for understanding and recognizing faces.

The method of combining decision trees and Gaussian kernel density estimation enables this algorithm to accurately model advanced features and complex relationships within facial data. This approach not only improves the accuracy of facial recognition but
also enhances the algorithm’s ability to generalize across different facial types and expressions. Through testing the algorithm on several standard datasets, we have observed that it achieves significantly higher recognition accuracy compared to traditional facial recognition techniques. While this algorithm has made significant strides in enhancing facial recognition accuracy, it does have its share of challenges and limitations. One example is the potential for higher computational costs due to the algorithm’s complexity, which could pose challenges when working with large-scale datasets. Future efforts will be directed toward enhancing the algorithm’s efficiency and reducing its computational resource requirements, making it more practical and efficient for a broad spectrum of application scenarios.

**Future Work**

The integration of decision trees and Gaussian kernel density estimation, while enhancing the accuracy of the Local Linear Embedding (LLE) algorithm, introduces several significant challenges. Firstly, the increased computational complexity may result in higher processing times and memory usage, posing limitations for large-scale datasets and real-time applications [44]. Secondly, the scalability of the algorithm is a concern, as performance may degrade with increasing dataset size, necessitating efficient scaling techniques to maintain accuracy [45]. Additionally, the sensitivity of decision trees and Gaussian kernel density estimation to noise and outliers can adversely impact the algorithm’s robustness and accuracy [46]. Lastly, while the algorithm demonstrates strong performance on standard facial recognition datasets, its generalization to more diverse and complex datasets requires further validation and adaptation to ensure broader applicability.

Future work will focus on addressing these limitations to enhance the algorithm’s practicality and efficiency. Optimization efforts will target reducing computational resource requirements by refining decision tree construction and Gaussian kernel density estimation processes, making the algorithm more suitable for large-scale and real-time applications. Improving scalability will involve designing more efficient algorithms and data structures to handle larger datasets without compromising accuracy or efficiency [47]. Enhancing robustness to noise and outliers could be achieved by incorporating robust statistical techniques or regularization methods. Implementing automated parameter tuning techniques, such as grid search or Bayesian optimization, will help identify optimal settings efficiently, ensuring consistent performance across different datasets. Extensive testing on a wider range of datasets will ensure generalization capability, adapting the algorithm to various real-world scenarios and data types, such as video sequences and 3D facial data. Additionally, integrating this algorithm with advanced techniques like deep learning frameworks could further boost performance, combining the strengths of different approaches for more accurate and efficient facial recognition systems. Through these future endeavors, we aim to advance the field of facial recognition and contribute to the development of more robust and efficient data analysis and classification techniques.

The integration of decision tree-based neighbor recognition and Gaussian kernel density estimation into the LLE algorithm significantly enhances its performance in various complex data processing tasks. This advanced algorithm has broad applications in fields such as healthcare and medical diagnostics, facial recognition and security, and autonomous vehicles, ultimately contributing to the improvement of human life.

**Author Contributions:** Conceptualization, H.G. and Z.Z.; methodology, H.G. and M.A.A.; software, H.G. and U.M.I.; validation, H.G., Z.Z., and J.O.; formal analysis, H.G.; investigation, Z.Q. and M.A.A.; resources, Z.Q. and U.M.I.; data curation, H.G.; writing—original draft preparation, H.G.; writing—review and editing, H.G.; visualization, H.G.; supervision, Z.Z.; project administration, H.G.; funding acquisition, Z.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the National Natural Science Foundation of China (No. 62006102).
Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References
32. Strehl, A. Relationship-Based Clustering and Cluster Ensembles for High-Dimensional Data Mining; The University of Texas at Austin: Austin, TX, USA, 2002.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.