# Analysis of Face Detection Techniques Using Artificial Intelligence

**Pragati[1] Minakshi Arora[2]**
[1]M.Tech. Scholar [2]Assistant Professor
[1,2]Department of Computer Science and Engineering
[1,2]SKITM, Bahadurgarh, India

*Abstract* — One of the most interesting problems in machine learning is face recognition. Face recognition is an issue that has been solved using a variety of strategies and approaches. In this research, we have demonstrated the effective usage of Principal Component Analysis in conjunction with the K Nearest Neighbors algorithm for face recognition. The K nearest neighbor algorithm is a non-parametric learning technique that determines the query point's final value by utilizing the target values of the K nearest data points. Eigen vectors are a notion used in PCA. An image is represented by an Eigen vector. K higher Eigen values are found by PCA to match to K Eigen vectors. Therefore, the PCA algorithm is a productive way to extract features for facial identification. Python is used as the programming language for implementation. This work illustrates the impact of combining the aforementioned technologies with their cutting-edge outcomes.

***Keywords:*** Eigen vectors, PCA, Face Recognition, KNN

## I. INTRODUCTION

The study of automatically recognizing or authenticating a person from an image or video sequences is known as face recognition [1]. In this research, we have demonstrated the effective usage of Principal Component Analysis in conjunction with the K Nearest Neighbors algorithm for face recognition. The K nearest neighbor algorithm is a non-parametric learning technique that determines the query point's final value by utilizing the target values of the K nearest data points. The class for the question point in a classification task will be the one with the highest frequency among the K data points; in a regression assignment, the target value for the query point will be the average of the target values of all K data points.

Principal Component Analysis, or PCA, addresses the curse of dimensionality by minimizing the amount of noise- and irrelevant-producing characteristics. The impact of the curse of dimensionality is seen in Fig. 1. Dimensions' highly connected aspects are eliminated by the application of a certain method. Real-time feature extraction and recognition for photos taken in advantageous (i.e., limited) conditions is now possible [2]. Eigen vectors are a notion used in PCA. An image is represented by an Eigen vector.

K higher Eigen values are found by PCA to match to K Eigen vectors. Therefore, the PCA algorithm is a productive way to extract features for facial identification. KNN combined with PCA can effectively address the face recognition problem.

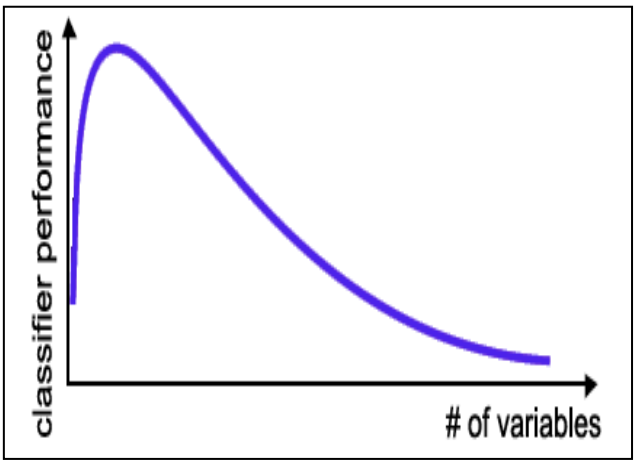


Fig. 1: Curse of Dimensionality

## II. PROPOSED METHODOLOGY:

We will now examine the applications of PCA and KNN to face recognition and other related problems. General workflow for face recognition is as:

1) A list of recognized faces
2) A database query is conducted and a new face is displayed.
3) The database yields a collection of faces that are most similar to the face that is being queried.
4) We select a face that most closely resembles the face in the query.

Face picture characteristics are extracted using the PCA approach. PCA projects the distinct features onto a lower dimensional feature and computes the covariance matrix's Eigenvectors. Eigen faces are another well-known name for these Eigenvectors [3]. Every image is represented as an Eigen vector using PCA. Take into consideration the 64 by 64 training image collection. Raster scan can be used to represent this training image as a vector of (4096 X 1). To obtain the mean normalized image, we deduct the mean image from the training image. From the database's accessible dataset, we obtain the mean image. We now have a vector of size (4096 X 1) that represents the mean normalized image. Let's say we have N training photos. In this case, each training image is represented as a normalized image by a matrix of size (4096 X N). This matrix should be M.

The covariance matrix is found by multiplying matrix M by its transpose. The size of the covariance matrix will be 4096 x 4096. Let's use R. R is now shown as:

$$R=P * D * PT$$

D: Diagonal matrix with Eigen values that has the same size as Matrix R

P: An orthonormal matrix with Eigen vectors that has the same size as Matrix R

PT: Orthonormal matrix (transpose of P) of the same size as Matrix R.

 

Eigen values for the covariance matrix are found once it has been obtained. Eigen values are arranged in decreasing order to generate a diagonal matrix D. Eigen vectors are now found, and they are used to build an orthonormal matrix. Choose P. Now we select K Eigen vectors corresponding to K highest Eigen values. So after selecting K Eigen vectors:

P1 : (4096 * K)

D1 : K * K

$P1^T$ : (K * 4096)

Now we do dimensionality reduction:

As we previously said, the size of our training images is 4096 * 4096. We multiply P1T by the mean normalized picture of a training image to obtain a reduced dimensional image.

(K * 4096) * (4096 * 1) = (K * 1).

In this manner, all training images are obtained as reduced dimensional images. After all training photos have been decreased in dimension, we can use the K-NN method to identify the face that is provided as the query point. Using a database of K features or dimensions, the K-NN algorithm selects the face with the highest matching score.

Euclidean distance or weighted Euclidean distance between features can be used for KNN.

e.g.

$X_i = \{x_{i1}, x_{i2}, x_{i3}, \ldots\ldots\ldots, x_{in}\}$

$X_j = \{x_{j1}, x_{j2}, x_{j3}, \ldots\ldots\ldots, x_{jn}\}$

Distance between these two vectors can be calculated as:

$$D(Xi, Xj) = \sqrt{\sum_{m=1}^{n} (\text{xim} - \text{xjm})^2}$$

In a similar manner, we can calculate the separation between each training data point and the goal data point. The K data points that are closest to the target data point can be chosen. To determine K nearest neighbors for target data points, more techniques exist. Finding a relationship between training characteristics or training data points can be done using techniques like the signal to noise ratio, Pearson coefficients, etc.

As we've seen, PCA primarily reduces dimensionality by identifying principal components as Eigen vectors. In the event of face recognition, we can then utilize KNN to identify the K data points or training photos that have a high degree of match with the target data point or image. The best matching image or data point can then be chosen from these K data points or training images. But there are other approaches as well; OpenCV, for example, is primarily utilized for these kinds of tasks. Intel created the OpenCV (Open Source Computer Vision suite), a suite of programming capabilities primarily geared toward real-time computer vision. It is a cross-platform library. Real-time image processing is its primary area of focus [4]. It is a classifier that facilitates the picture processing. Image processing is the primary topic and can be applied to the newest methods [5].

## III. Implementation using Python (SCIKIT learn):

Thus far, we have observed that face identification using KNN and PCA is possible when training data is provided as a vector of photos. Each training set of data represents a feature, which is essentially the image's pixel value. Feature reduction is done by PCA, and target picture classification is assisted by KNN. We can utilize Python to carry out these activities. The Python programming language has several applications in data science. Python can be used to tackle data science activities such as data analysis and visualization, which are essential components of data science. Python has a large number of libraries that can be used to accomplish this kind of work [6].

The steps for implementation are as follows:

1) The data can be divided into training and test data using the train test split function from sklearn.cross validation. Typically, we divide the test data into 25% and the training data into 75%.
2) Eigen face computation, or Eigen vector computation In order to convert training and test data into principle components, Eigen vectors of a given size can be found using randomized PCA from sklearn.decomposition.
3) Locate the closest neighbors and observe how the algorithm operates. The sklearn KNeighbors Classifier.neighbors can be used to determine the necessary number of closest neighbors and thereafter see how the algorithm operates.

Python is a very effective language for working with all types of data, including text, photos, videos, and more. Python scripting is an interpreted, general-purpose, high-level programming language that is available as freeware. Code readability is prioritized in design philosophy [7]. Numerous packages for data analysis, data visualization, and other uses are included in it. In machine learning projects, we often preprocess the data first, then use the provided data to construct a model that meets the requirements. We next visualize the model's operation using graphs, charts, etc. to determine its correctness. Python is a programming language that can be used for all of these activities. Python is a programming language that may be used for a wide range of tasks, from data pretreatment to result visualization.

## IV. Conclusion:

Here, we can see how effectively faces can be recognized using the K-NN and PCA algorithms. It is a more beneficial strategy even with the use of Python. When we train a model based on instances rather than directly on classes, PCA, an instance-based learning approach, is very effective. But, in order to achieve better results when it comes to feature extraction or feature reduction, we can apply an alternative algorithm in place of PCA. To achieve better results, we can apply different values of K for different types of data sets. Even neural networks are producing fascinating categorization results. Neural networks can also be used for facial recognition.

## References:

[1] Mehran Kafai, Member, IEEE, Le An, Student Member, IEEE, and Bir Bhanu, Fellow, IEEE, "Reference Face Graph for Face Recognition", IEEE, ISSN - 1556-6013 , 2022.

[2] Kavita , Ms. Manjeet Kaur," A Survey paper for Face Recognition Technologies", International Journal of

 

Scientific and Research Publications, ISSN 2250-3153, Volume 6, Issue 7, July 2020.
[3] Ashutosh Chandra Bhensle, Rohit Raja," An Efficient Face Recognition using PCA and Euclidean Distance Classification", IJCSMC, ISSN 2320–088X, Vol. 3, Issue. 6, June 221.
[4] P Y kumbhar , Mohammad attaullah , Shubham Dhere , Shivkumar Hipparagi, "Real time face detection and tracking using OpenCV", International Journal for Research in Emerging Science and Technology, ISSN: 2349-7610, Volume-4, Issue-4, Apr-2019.
[5] Manik Sharma, J Anuradha, H K Manne and G S C Kashyap, "Facial detection using deep learning", IOP Publishing, 14th ICSET, 2018.
[6] Gaurav, Ritu Sindhu, "Python as a key for data science", IJCSE, ISSN-2347-2693, Volume-6, Issue-4, Apr-2018.
[7] Ing. Zdena Dobesova, "Programming Language Python for Data Processing", IEEE, International Conference on Electrical and Control Engineering (ICECE), ISBN: 978-1-4244-8165-1, 2017 .



 