**Fingerprint Identification with Fusion of Gabor and Minutiae Features Using BPNN Classifier**

**ABSTRACT**

This study presents an innovative approach to fingerprint identification by leveraging the fusion of Gabor and Minutiae features, employing a Backpropagation Neural Network (BPNN) classifier for accurate categorization. The process involves initial handling of a fingerprint image dataset, including crucial pre-processing steps such as resizing images and implementing morphological operations like dilation, erosion, and opening. Subsequently, Gabor features and minutiae extraction are performed, followed by the fusion of these features to create a comprehensive representation. To address dimensionality concerns, Principal Component Analysis (PCA) is applied. The dataset, comprising the fused features and corresponding labels, is then loaded for the final step - classification using a BPNN. The network is configured for feed-forward backpropagation, distinguishing fingerprint patterns into categories such as arch, left loop, right loop, tented, and whorl. The evaluation metric used to measure the success of the classification process is accuracy. This approach aims to enhance fingerprint recognition by combining distinctive Gabor and minutiae features, ultimately achieving a more robust and precise identification system through the utilization of neural network-based classification.

**Keywords**: Finger print images dataset, BPNN, Deep learning techniques, PCA, classification and pre-processing, Minutiae features.

**CHAPTER 1**

**INTRODUCTION**

This study focuses on advancing fingerprint identification through the application of Convolutional Neural Networks (CNNs), a prominent deep learning technique. Fingerprint identification plays a pivotal role in various domains such as law enforcement, access control, and immigration. Traditional methods of fingerprint analysis have relied on manual interpretation and feature extraction, which can be time-consuming and prone to error. By employing CNNs, which are capable of learning intricate patterns and features directly from raw data, this research seeks to enhance the accuracy and efficiency of fingerprint identification. The classification task involves categorizing fingerprints into distinct patterns, including Arch, Left Loop, Right Loop, Tented, and Whorl. CNNs offer significant advantages in handling complex data structures like fingerprint images, making them well-suited for this classification task. The utilization of CNNs in fingerprint identification holds promise for improving biometric security systems, streamlining identification processes, and enhancing overall security measures.

Fingerprints are unique patterns, made by friction ridges (raised) and furrows (recessed), which appear on the pads of the fingers and thumbs. Prints from palms, toes and feet are also unique; however, these are used less often for identification, so this guide focuses on prints from the fingers and thumbs.

The fingerprint pattern, such as the print left when an inked finger is pressed onto paper, is that of the friction ridges on that particular finger. Friction ridge patterns are grouped into three distinct types—loops, whorls, and arches—each with unique variations, depending on the shape and relationship of the ridges:

**Loops** - prints that recurve back on themselves to form a loop shape. Divided into radial loops (pointing toward the radius bone, or thumb) and ulnar loops (pointing toward the ulna bone, or pinky), loops account for approximately 60 percent of pattern types.

The Left Loop Pattern fingerprint is a unique identifier used in forensic analysis of fingerprints. It refers to a specific arrangement of ridges in the fingerprint pattern, where the loop pattern leans towards the left side. This distinctive characteristic aids in identifying individuals in criminal investigations and other security-related matters. By analysing the minutiae and orientation of these loops, forensic experts can match fingerprints to individuals with a high degree of accuracy, providing crucial evidence in solving crimes and establishing identities.



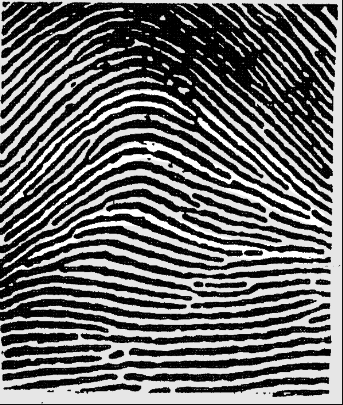
A "Right Loop Pattern" is a distinctive fingerprint pattern characterized by the looping of ridges towards the right side of the finger. It's a common classification in forensic fingerprint analysis, aiding in identifying individuals based on unique ridge patterns. These loops are crucial in forensic investigations, helping to differentiate between individuals and contributing to the reliability of fingerprint evidence in criminal cases. Understanding these patterns involves intricate analysis of ridge formations, their directionality, and minutiae points, enabling forensic experts to accurately match fingerprints to individuals and provide crucial evidence in legal proceedings.



**Whorls** - form circular or spiral patterns, like tiny whirlpools. There are four groups of whorls: plain (concentric circles), central pocket loop (a loop with a whorl at the end), double loop (two loops that create an S-like pattern) and accidental loop (irregular shaped). Whorls make up about 35 percent of pattern types.

Whorls are one of the primary pattern types found in fingerprints, characterized by circular or spiral formations. These distinctive patterns play a crucial role in forensic science for identification purposes due to their uniqueness to each individual. Within the field of fingerprint analysis, experts utilize whorl patterns as key identifiers in criminal investigations and personal identification processes. Understanding the intricacies of whorl patterns aids in accurate classification and comparison of fingerprints, contributing significantly to law enforcement efforts worldwide.

**Arches** - create a wave-like pattern and include plain arches and tented arches. Tented arches rise to a sharper point than plain arches. Arches make up about five percent of all pattern types.

Arch Pattern fingerprinting is a method used in cybersecurity to identify and categorize software architecture patterns within applications. It involves analyzing the structure and behavior of software to recognize recurring design motifs or frameworks. By studying these fingerprints, analysts can better understand the underlying architecture of an application, aiding in tasks such as vulnerability assessment, code optimization, and system compatibility checks. This process helps organizations in assessing the security posture of their software systems, detecting potential weaknesses, and making informed decisions to mitigate risks effectively.

**Tented**: The tented pattern fingerprint is a distinctive ridge pattern found in forensic analysis. Characterized by ridges converging towards a central point, resembling the peak of a tent, it is one of several types identified in fingerprint identification. This unique pattern plays a crucial role in forensic investigations, aiding in the accurate identification of individuals. Its distinctiveness stems from the arrangement and orientation of ridges, allowing forensic experts to differentiate between individuals based on their fingerprint patterns. Understanding and analyzing tented pattern fingerprints are fundamental in criminal investigations, providing valuable evidence in solving cases.

Fingerprints, a hallmark of human identity, have long fascinated scientists and law enforcement alike. These unique patterns, formed during fetal development and unchanged throughout life, serve as a distinct identifier for individuals. By analyzing the intricate ridges, loops, and whorls, forensic experts can link suspects to crime scenes with remarkable precision. The study of fingerprints, known as dermatoglyphics, encompasses not only their classification but also their evolutionary significance and forensic applications. From ancient civilizations to modern biometric security systems, fingerprints remain a cornerstone of identification, unlocking doors to both historical mysteries and contemporary crime investigations.

**CHAPTER 2**

**LITERATURE SURVEY**

**[1] Hasan H, Abdul-Kareem S.** Fingerprint image enhancement and recognition algorithms: a survey. Neural Comput Appl 2013;(23):1605–10.

Fingerprint systems have received a great deal of research and attracted many researchers’ effort since they provide a powerful tool for access control and security and for practical applications. A literature review of the techniques used to extract the features of fingerprint as well as recognition techniques is given in this paper. Some of the reviewed research articles have used traditional methods such as recognition techniques, whereas the other articles have used neural networks methods. In addition, fingerprint techniques of enhancement are introduced.

**Summary:** Survey of algorithms for enhancing and recognizing fingerprint images.

**[2]** **Win KN, Li K, Chen J, Viger PF, Li K.** Fingerprint classification and identification algorithms for criminal investigation: a survey. Fut Gener Comput Syst 2020; 110:758–71.

Fingerprint plays a fundamental role in community security and criminal investigation, such as [forensic investigation](https://www.sciencedirect.com/topics/computer-science/forensic-investigation), law enforcement, customs access and public security organs. This can also help to provide a more enjoyable and secure life to people. Various [machine learning](https://www.sciencedirect.com/topics/computer-science/machine-learning) and [neural network approaches](https://www.sciencedirect.com/topics/computer-science/neural-network-approach) have been proposed for fingerprint acquisition, detection, classification, and analysis. In this survey, we present an up-to-date literature evaluation of fingerprint [classification algorithms](https://www.sciencedirect.com/topics/computer-science/classification-algorithm) and fingerprint application in the area of criminal investigation. Firstly, we discuss the characteristics of fingerprint and the application in criminal investigation. In addition, we analyze and compare [machine learning algorithms](https://www.sciencedirect.com/topics/computer-science/machine-learning-algorithm) of fingerprint in terms of classification, matching, feature extraction, fingerprint and finger-vein recognition, and spoof detection. Finally, we highlight the challenges in the fingerprint analysis and discuss the future directions.

**Summary:** Survey explores fingerprint algorithms for crime scene analysis and identification.

**[3**] **AlShehri H, Hussain M, AboAlSamh H, AlZuair M.** A large-scale study of fingerprint matching systems for sensor interoperability problem. Sensors 2018;18: 1008.

The fingerprint is a commonly used biometric modality that is widely employed for authentication by law enforcement agencies and commercial applications. The designs of existing fingerprint matching methods are based on the hypothesis that the same sensor is used to capture fingerprints during enrollment and verification. Advances in fingerprint sensor technology have raised the question about the usability of current methods when different sensors are employed for enrollment and verification; this is a fingerprint sensor interoperability problem. To provide insight into this problem and assess the status of state-of-the-art matching methods to tackle this problem, we first analyze the characteristics of fingerprints captured with different sensors, which makes cross-sensor matching a challenging problem. We demonstrate the importance of fingerprint enhancement methods for cross-sensor matching. Finally, we conduct a comparative study of state-of-the-art fingerprint recognition methods and provide insight into their abilities to address this problem. We performed experiments using a public database (FingerPass) that contains nine datasets captured with different sensors. We analysed the effects of different sensors and found that cross-sensor matching performance deteriorates when different sensors are used for enrollment and verification. In view of our analysis, we propose future research directions for this problem.

**Summary:** Study explores fingerprint systems for sensor compatibility in large-scale operations.

**[4] Shemmary ENAAl.** Classification of fingerprint images using neural networks technique. J Eng (JOE) 2012;1(3):40–8.

The uniqueness, public recognition, firmness, and their least jeopardy of fingerprints made an extensively and proficiently utilized personal [authentication](https://www.sciencedirect.com/topics/computer-science/authentication) metrics. Fingerprint technology is a [biometric](https://www.sciencedirect.com/topics/computer-science/biometrics) method that is used to recognize persons on the basis of their physical traits. These physical forms comprise of ridges and valleys prevailing on the surface of fingertips. [Fingerprint images](https://www.sciencedirect.com/topics/computer-science/fingerprint-image) are direction-oriented pattern fashioned using ridges and valleys. The reputation of the fingerprint image regulates the durability of a fingerprint [authentication scheme](https://www.sciencedirect.com/topics/computer-science/authentication-scheme). For enhancing the restrictions of prevailing fingerprint image augmentation approaches we have proposed an effectual method to pact with various fingerprint images. The proposed methodology alienated into three modules. Primarily, the fingerprint image is endangered to [denoising](https://www.sciencedirect.com/topics/computer-science/de-noising) procedure where Wave atom transform is used. Once this procedure is accomplished the image augmentation is achieved for improving the classification rate. The [morphological operation](https://www.sciencedirect.com/topics/computer-science/morphological-operations) is used in our proposed technique in order to augment the image. The [morphological operators](https://www.sciencedirect.com/topics/computer-science/morphological-operator) such as dilation and area opening are used here for improvement. Finally, the ordering of fingerprint image is done. Adaptive Genetic [Neural Network](https://www.sciencedirect.com/topics/computer-science/neural-network) (AGNN) is used for classification of images efficiently.

**Summary:** Fingerprint image classification using neural networks for identification.

**[5] Leung KC, Leung CH.** Improvement of fingerprint retrieval by a statistical classifier. IEEE Trans Inform Forens Secur 2011;6(1):59–69.

The topics of fingerprint classification, indexing, and retrieval have been studied extensively in the past decades. One problem faced by researchers is that in all publicly available fingerprint databases, only a few fingerprint samples from each individual are available for training and testing, making it inappropriate to use sophisticated statistical methods for recognition. Hence most of the previous works resorted to simple k -nearest neighbors ( k -NN) classification. However, the k -NN classifier has the drawbacks of being comparatively slow and less accurate. In this paper, we tackle this problem by first artificially expanding the set of training samples using our previously proposed spatial modeling technique. With the expanded training set, we are then able to employ a more sophisticated classifier such as the Bayes classifier for recognition. We apply the proposed method to the problem of one-to- N fingerprint identification and retrieval. The accuracy and speed are evaluated using the benchmarking FVC 2000, FVC 2002, and NIST-4 databases, and satisfactory retrieval performance is achieved.

**Summary:** Enhancing fingerprint matching with advanced statistical analysis.

**[6] Wan GC, Li MM, Xu H, Kang WH, Rui JW, Tong MS.** X-Finger-net: pixel-wise segmentation method for partially defective fingerprint based on attention gates and U-net. Sensors 2020; 20:4473.

Partially defective fingerprint image (PDFI) with poor performance poses challenges to the automated fingerprint identification system (AFIS). To improve the quality and the performance rate of PDFI, it is essential to use accurate segmentation. Currently, most fingerprint image segmentations use methods with ridge orientation, ridge frequency, coherence, variance, local gradient, etc. This paper proposes a method of X-Finger-Net for segmenting PDFIs. Based on U-Net, X-Finger-Net inherits its characteristics. The attention gate with fewer parameters is used to replace the cascaded network, which can suppress uncorrelated regions of PDFIs. Moreover, the X-Finger-Net implements a pixel-level segmentation and takes non-blocking fingerprint images as an input to preserve the global characteristics of PDFIs. The X-Finger-Net can achieve a very good segmentation effect as demonstrated in the self-made fingerprint segmentation test.

**Summary:** X-Finger-net: Accurate pixel segmentation for flawed fingerprints.

**CHAPTER 3**

**EXISTING METHOD**

**Block Diagrams:**

Input Image

Pre - Processing

PCA

Convolution Neural Network

Layers

Training Options

Classification

Image Resize

Morphological Operations

* Dilate Image
* Erode Image
* Opening of Image

Accuracy

Gabor Features

Classification

* Arch
* Left Loop
* Right Loop
* Tented
* Whorl

**Figure 1. Existing Method Block Diagram**

The Existing Method involves leveraging Gabor features and a Convolutional Neural Network (CNN) classifier to enhance fingerprint identification. Initially, the fingerprint images undergo preprocessing steps, including resizing, morphological operations, and extraction of Gabor phase and texture features. These preprocessing techniques aim to standardize image sizes, enhance feature extraction, and improve the quality of fingerprint representations. Principal Component Analysis (PCA) is then utilized for dimensionality reduction, optimizing feature selection and improving computational efficiency. Subsequently, the CNN classifier, powered by deep learning techniques, is employed for classification. The CNN model is trained using labelled data to recognize various fingerprint patterns, such as Arch, Left Loop, Right Loop, Tented, and Whorl. Through the training process, the CNN learns to differentiate between different fingerprint patterns based on the extracted Gabor features. The classification accuracy of the system is evaluated to gauge its effectiveness in accurately categorizing fingerprints into their respective patterns. By integrating Gabor features with CNN classification, this methodology aims to achieve heightened accuracy in fingerprint identification, thereby advancing the capabilities of biometric security systems.

**Pre-Processing:**

Preprocessing plays a crucial role in enhancing the quality of fingerprint images and facilitating accurate feature extraction. It begins with input fingerprint images, which may vary in size and quality. Resizing these images to a standardized dimension not only ensures consistency but also reduces computational complexity. Morphological operations are then applied to refine the fingerprint ridges and valleys, enhancing their clarity and continuity. Next, Gabor filters are utilized to extract both phase and texture features from the fingerprint image. Gabor phase features capture the orientation and direction of ridges, while Gabor texture features characterize the fine details and patterns within the ridges. These Gabor features provide a robust representation of the fingerprint's unique characteristics, enabling more accurate identification. By combining both phase and texture features, the preprocessing stage aims to create a comprehensive fingerprint representation that captures essential details while minimizing noise and irrelevant information. Overall, preprocessing lays the foundation for effective feature extraction and classification, ultimately improving the accuracy and reliability of fingerprint identification systems.

**Image Resize:**

Resizing an image in MATLAB involves altering its dimensions while preserving its aspect ratio or adjusting it as desired. The process is pivotal in various image processing tasks, including preprocessing for analysis or visualization. MATLAB offers versatile functions like "imresize" for resizing images conveniently. Upon calling this function, users specify the input image and the desired dimensions for the output image. MATLAB then employs interpolation techniques to resample pixel values, effectively adjusting the image's size. Bilinear or bicubic interpolation methods are commonly used to maintain image quality during resizing. Bilinear interpolation estimates new pixel values based on nearby pixels, while bicubic interpolation considers a larger neighbourhood for a smoother result. It's essential to note that resizing may lead to loss of information or aliasing if the new dimensions are significantly smaller than the original image. Conversely, enlarging an image can result in interpolation artifacts or blurring if the resolution is increased beyond the original. Thus, selecting appropriate interpolation methods and considering the specific requirements of the application are crucial for achieving desired results. In summary, MATLAB's resizing capabilities provide a flexible and efficient solution for adjusting image sizes while maintaining or enhancing image quality, supporting a wide range of image processing tasks.

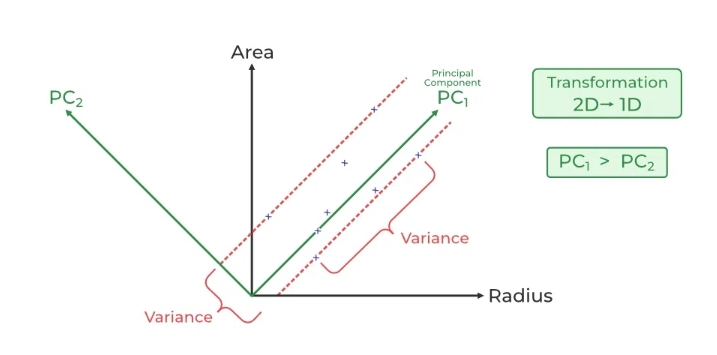
**Morphological Operations:**

Morphological operations in MATLAB are fundamental image processing techniques used to manipulate and analyze the structure of images based on their shape and form. These operations primarily involve the dilation and erosion of images, which are essential for tasks like noise removal, edge detection, and feature extraction. Dilation expands the boundaries of objects in an image, effectively thickening them, while erosion shrinks the object boundaries, making them thinner. These operations are often applied iteratively to enhance or refine specific features within an image. Additionally, MATLAB provides functions for other morphological operations such as opening, closing, and boundary extraction, which further aid in image enhancement and segmentation. Opening involves the sequential application of erosion followed by dilation and is commonly used for removing small objects and smoothing contours. Closing, on the other hand, performs dilation followed by erosion and is useful for filling small gaps and restoring object shapes. Boundary extraction isolates the outlines of objects in an image, facilitating subsequent analysis or feature extraction tasks. Overall, morphological operations in MATLAB offer powerful tools for preprocessing and analyzing images, contributing to various applications in fields such as medical imaging, computer vision, and remote sensing.

**Gabor phase and Gabor texture Features)**

Gabor phase and texture features are essential components in fingerprint analysis, particularly in MATLAB implementations. Gabor filters are mathematical functions used to extract features by convolving them with the input fingerprint image. Gabor phase features capture the orientation and direction of ridges in the fingerprint, providing valuable information about the overall structure. These features are computed by taking the phase component of the convolved image, which represents the local ridge orientation. Gabor texture features, on the other hand, characterize the fine details and patterns within the fingerprint ridges. This is achieved by considering the magnitude component of the convolved image, which highlights variations in the intensity or contrast of the ridges. In MATLAB, implementing Gabor phase and texture features typically involves defining parameters such as the frequency, orientation, and aspect ratio of the Gabor filters. These parameters influence the scale and sensitivity of the features extracted, allowing for customization based on the specific requirements of the fingerprint analysis task. Overall, leveraging MATLAB for Gabor feature extraction enables researchers and practitioners to efficiently process fingerprint images and extract discriminative features that are crucial for accurate identification and classification.

**PCA:**

****

As the number of features or dimensions in a dataset increases, the amount of data required to obtain a statistically significant result increase exponentially. This can lead to issues such as overfitting, increased computation time, and reduced accuracy of machine learning models this is known as the curse of dimensionality problems that arise while working with high-dimensional data.

As the number of dimensions increases, the number of possible combinations of features increases exponentially, which makes it computationally difficult to obtain a representative sample of the data and it becomes expensive to perform tasks such as clustering or classification because it becomes. Additionally, some [machine learning](https://www.geeksforgeeks.org/machine-learning/) algorithms can be sensitive to the number of dimensions, requiring more data to achieve the same level of accuracy as lower-dimensional data.

To address the [curse of dimensionality](https://www.geeksforgeeks.org/videos/curse-of-dimensionality-in-machine-learning/), [Feature engineering](https://www.geeksforgeeks.org/what-is-feature-engineering/)techniques are used which include feature selection and feature extraction. [Dimensionality reduction](https://www.geeksforgeeks.org/dimensionality-reduction/) is a type of feature extraction technique that aims to reduce the number of input features while retaining as much of the original information as possible.

**What is Principal Component Analysis (PCA)?**

[Principal Component Analysis](https://www.geeksforgeeks.org/principal-component-analysis-with-python/)(PCA) technique was introduced by the mathematician **Karl Pearson** in 1901**.** It works on the condition that while the data in a higher dimensional space is mapped to data in a lower dimension space, the variance of the data in the lower dimensional space should be maximum.

**Principal Component Analysis (PCA)**is a statistical procedure that uses an orthogonal transformation that converts a set of correlated variables to a set of uncorrelated variables. PCA is the most widely used tool in exploratory data analysis and in machine learning for predictive models. Moreover,

Principal Component Analysis (PCA) is an [unsupervised learning](https://www.geeksforgeeks.org/supervised-unsupervised-learning/) algorithm technique used to examine the interrelations among a set of variables. It is also known as a general factor analysis where regression determines a line of best fit.

The main goal of Principal Component Analysis (PCA) is to reduce the dimensionality of a dataset while preserving the most important patterns or relationships between the variables without any prior knowledge of the target variables.

Principal Component Analysis (PCA) is used to reduce the dimensionality of a data set by finding a new set of variables, smaller than the original set of variables, retaining most of the sample’s information, and useful for the [regression and classification](https://www.geeksforgeeks.org/regression-classification-supervised-machine-learning/) of data.

* Principal Component Analysis (PCA) is a technique for dimensionality reduction that identifies a set of orthogonal axes, called principal components, that capture the maximum variance in the data. The principal components are linear combinations of the original variables in the dataset and are ordered in decreasing order of importance. The total variance captured by all the principal components is equal to the total variance in the original dataset.
* The first principal component captures the most variation in the data, but the second principal component captures the maximum [variance](https://www.geeksforgeeks.org/python-statistics-variance/) that is [orthogonal](https://www.geeksforgeeks.org/orthogonal-and-orthonormal-vectors-in-linear-algebra/) to the first principal component, and so on.
* Principal Component Analysis can be used for a variety of purposes, including data visualization, feature selection, and data compression. In data visualization, PCA can be used to plot high-dimensional data in two or three dimensions, making it easier to interpret. In feature selection, PCA can be used to identify the most important variables in a dataset. In data compression, PCA can be used to reduce the size of a dataset without losing important information.
* In Principal Component Analysis, it is assumed that the information is carried in the variance of the features, that is, the higher the variation in a feature, the more information that features carries.

**Training Options:**

1. **sgdm (Stochastic Gradient Descent with Momentum):** Specifies the optimization algorithm for updating the neural network weights during training.
2. **ExecutionEnvironment: "auto":** Automatically selects the available hardware (CPU or GPU) for training the neural network, optimizing computation resources.
3. **InitialLearnRate: 0.001:** Sets the initial learning rate for the optimization algorithm, determining the step size in weight updates during training.
4. **MaxEpochs: 100:** Specifies the maximum number of training epochs, i.e., the number of passes through the entire training dataset.
5. **MiniBatchSize: 25:** Defines the number of samples in each mini-batch, controlling the frequency of weight updates and impacting training efficiency.
6. **Shuffle: "every-epoch”:** Specifies that the training data should be shuffled before each epoch, preventing the model from memorizing the order of the data.
7. **ValidationData: augimdsValidation:** Sets the validation dataset used to monitor the model's performance during training, aiding in preventing overfitting.
8. **ValidationFrequency: 50:** Determines how often to evaluate the validation set, here after every 50 mini-batches.
9. **Verbose: true:** Enables the display of detailed information about the training progress.
10. **Plots: "training-progress":** Requests the generation of training progress plots, providing visual insights into the model's performance over epochs.

**Convolutional Neural Network:**

Convolutional Neural Networks (CNNs) stand as a cornerstone in the realm of deep learning, revolutionizing various fields, particularly computer vision, with their unparalleled ability to comprehend and extract intricate patterns from visual data. In MATLAB, CNNs have garnered significant attention due to their exceptional capacity to process complex image-based information.

At its core, a CNN mimics the visual cortex's architecture, composed of multiple layers that progressively learn and abstract hierarchical features from raw pixel inputs. This hierarchical feature extraction enables CNNs to automatically detect essential details like edges, textures, and shapes, making them highly adept at tasks such as image classification, object detection, and image segmentation.

Within MATLAB's environment, constructing and training CNNs is remarkably accessible, aided by the deep learning toolbox that provides pre-defined layers, training functions, and visualization tools. The versatility of MATLAB further facilitates data preprocessing, augmentation, and integration of custom architectures, empowering researchers and practitioners to tailor CNNs to diverse applications. This introduction delves into the significance of Convolutional Neural Networks in MATLAB, highlighting their role in pushing the boundaries of computer vision capabilities. As the synergy between deep learning and MATLAB advances, CNNs continue to fuel breakthroughs, influencing everything from medical diagnostics to autonomous vehicles, underlining their profound impact on modern technological landscapes.

The network architecture can vary depending on the types and numbers of layers included. The types and number of layers included depends on the particular application or data. For example, classification networks typically have a softmax layer and a classification layer, whereas regression networks must have a regression layer at the end of the network. A smaller network with only one or two convolutional layers might be sufficient to learn on a small number of grayscale image data. On the other hand, for more complex data with millions of colored images, you might need a more complicated network with multiple convolutional and fully connected layers.

**Image Input Layer:**

The Image Input Layer in MATLAB plays a pivotal role in seamlessly integrating image data into various deep learning workflows. Serving as the entry point of the neural network architecture, this layer facilitates the ingestion of image data with diverse dimensions and formats. It accommodates the preprocessing and normalization steps necessary for model training, enabling the network to efficiently learn essential features. Additionally, the Image Input Layer offers flexibility in handling augmented datasets, contributing to improved model robustness. With its user-friendly interface and compatibility with various neural network architectures, this layer significantly simplifies the process of incorporating image data into MATLAB-based deep learning pipelines.

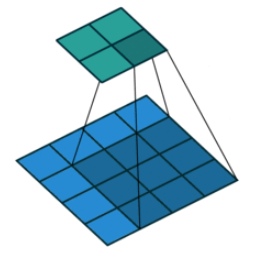
**Convolutional Layer:**

A 2-D convolutional layer applies sliding convolutional filters to 2-D input. Create a 2-D convolutional layer using convolution2dLayer. The convolutional layer consists of various components.

* **Filters and Stride**

A convolutional layer consists of neurons that connect to subregions of the input images or the outputs of the previous layer. The layer learns the features localized by these regions while scanning through an image. When creating a layer using the convolution2dLayer function, you can specify the size of these regions using the filter Size input argument.

For each region, the trainNetwork function computes a dot product of the weights and the input, and then adds a bias term. A set of weights that is applied to a region in the image is called a filter. The filter moves along the input image vertically and horizontally, repeating the same computation for each region. In other words, the filter convolves the input. This image shows a 3-by-3 filter scanning through the input. The lower map represents the input and the upper map represents the output.

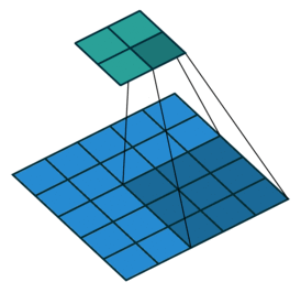


**Fig: Filters and Stride**

The step size with which the filter moves is called a stride. You can specify the step size with the Stride name-value pair argument. The local regions that the neurons connect to can overlap depending on the filterSize and 'Stride' values.

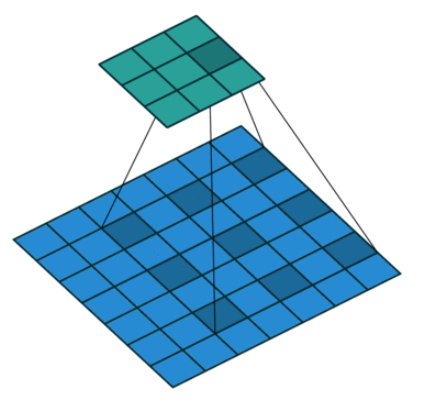
This image shows a 3-by-3 filter scanning through the input with a stride of 2. The lower map represents the input and the upper map represents the output.

The number of weights in a filter is h \* w \* c, where h is the height, and w is the width of the filter, respectively, and c is the number of channels in the input. For example, if the input is a color image, the number of color channels is 3. The number of filters determines the number of channels in the output of a convolutional layer. Specify the number of filters using the Num Filters argument with the convolution2dLayer function.



* **Dilated Convolution**

A dilated convolution is a convolution in which the filters are expanded by spaces inserted between the elements of the filter. Specify the dilation factor using the 'DilationFactor' property. Use dilated convolutions to increase the receptive field (the area of the input which the layer can see) of the layer without increasing the number of parameters or computation.



**Dilated Convolution**

The layer expands the filters by inserting zeros between each filter element. The dilation factor determines the step size for sampling the input or equivalently the up-sampling factor of the filter. It corresponds to an effective filter size of (*Filter Size* – 1). \* *Dilation Factor* + 1. For example, a 3-by-3 filter with the dilation factor [2 2] is equivalent to a 5-by-5 filter with zeros between the elements.

This image shows a 3-by-3 filter dilated by a factor of two scanning through the input. The lower map represents the input and the upper map represents the output.

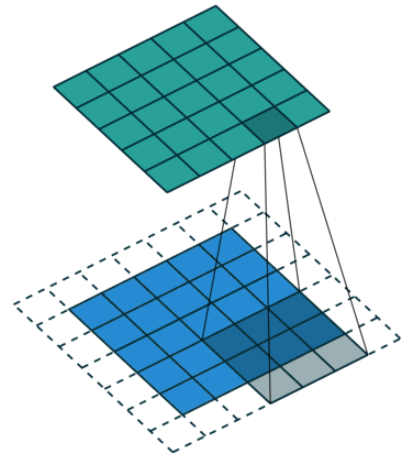
* **Feature Maps**

As a filter moves along the input, it uses the same set of weights and the same bias for the convolution, forming a feature map. Each feature map is the result of a convolution using a different set of weights and a different bias. Hence, the number of feature maps is equal to the number of filters. The total number of parameters in a convolutional layer is ((h\*w\*c + 1) \*Number of Filters), where 1 is the bias.

#### Padding

You can also apply padding to input image borders vertically and horizontally using the 'Padding' name-value pair argument. Padding is values appended to the borders of the input to increase its size. By adjusting the padding, you can control the output size of the layer.

This image shows a 3-by-3 filter scanning through the input with padding of size 1. The lower map represents the input and the upper map represents the output.



**Fig: Padding**

#### Output Size

The output height and width of a convolutional layer is (Input Size – ((Filter Size – 1) \*Dilation Factor + 1) + 2\*Padding)/Stride + 1. This value must be an integer for the whole image to be fully covered. If the combination of these options does not lead the image to be fully covered, the software by default ignores the remaining part of the image along the right and bottom edges in the convolution.

#### Number of Neurons

The product of the output height and width gives the total number of neurons in a feature map, say *Map Size*. The total number of neurons (output size) in a convolutional layer is *Map Size*\*Number *of* Filters.

For example, suppose that the input image is a 32-by-32-by-3 color image. For a convolutional layer with eight filters and a filter size of 5-by-5, the number of weights per filter is 5 \* 5 \* 3 = 75, and the total number of parameters in the layer is (75 + 1) \* 8 = 608. If the stride is 2 in each direction and padding of size 2 is specified, then each feature map is 16-by-16. This is because (32 – 5 + 2 \* 2)/2 + 1 = 16.5, and some of the outermost padding to the right and bottom of the image is discarded. Finally, the total number of neurons in the layer is 16 \* 16 \* 8 = 2048.

Usually, the results from these neurons pass through some form of nonlinearity, such as rectified linear units (ReLU).

#### Learning Parameters

You can adjust the learning rates and regularization options for the layer using name-value pair arguments while defining the convolutional layer. If you choose not to specify these options, then trainNetwork uses the global training options defined with the training Options function.

#### Number of Layers

A convolutional neural network can consist of one or multiple convolutional layers. The number of convolutional layers depends on the amount and complexity of the data.

**Batch Normalization Layer**

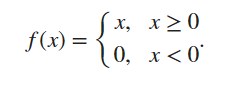
Create a batch normalization layer using batchNormalizationLayer. A batch normalization layer normalizes a mini-batch of data across all observations for each channel independently. To speed up training of the convolutional neural network and reduce the sensitivity to network initialization, use batch normalization layers between convolutional layers and nonlinearities, such as ReLU layers. The layer first normalizes the activations of each channel by subtracting the mini-batch mean and dividing by the mini-batch standard deviation. Then, the layer shifts the input by a learnable offset β and scales it by a learnable scale factor γ. β and γ are themselves learnable parameters that are updated during network training.

Batch normalization layers normalize the activations and gradients propagating through a neural network, making network training an easier optimization problem. To take full advantage of this fact, you can try increasing the learning rate. Since the optimization problem is easier, the parameter updates can be larger and the network can learn faster. You can also try reducing the L2 and dropout regularization.

With batch normalization layers, the activations of a specific image during training depend on which images happen to appear in the same mini-batch. To take full advantage of this regularizing effect, try shuffling the training data before every training epoch. To specify how often to shuffle the data during training, use the 'Shuffle' name-value pair argument of training Options.

**ReLU Layer**

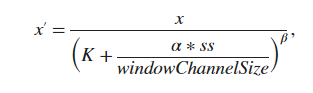
Create a ReLU layer using reluLayer. A ReLU layer performs a threshold operation to each element of the input, where any value less than zero is set to zero. Convolutional and batch normalization layers are usually followed by a nonlinear activation function such as a rectified linear unit (ReLU), specified by a ReLU layer. A ReLU layer performs a threshold operation to each element, where any input value less than zero is set to zero, that is,

****

The ReLU layer does not change the size of its input. There are other nonlinear activation layers that perform different operations and can improve the network accuracy for some applications. For a list of activation layers, see Activation Layers.

**Cross Channel Normalization (Local Response Normalization) Layer**

Create a cross channel normalization layer using crossChannelNormalizationLayer. A channel-wise local response (cross-channel) normalization layer carries out channel-wise normalization. This layer performs a channel-wise local response normalization. It usually follows the ReLU activation layer. This layer replaces each element with a normalized value it obtains using the elements from a certain number of neighbouring channels (elements in the normalization window). That is, for each element x in the input, trainNetwork computes a normalized value x ′ Using



where K, α, and β are the hyperparameters in the normalization, and ss is the sum of squares of the elements in the normalization window. You must specify the size of the normalization window using the windowChannelSize argument of the crossChannelNormalizationLayer function. You can also specify the hyperparameters using the Alpha, Beta, and K name-value pair arguments. The previous normalization formula is slightly different than what is presented. You can obtain the equivalent formula by multiplying the alpha value by the windowChannelSize.

**Max and Average Pooling Layers**

A 2-D max pooling layer performs downsampling by dividing the input into rectangular pooling regions, then computing the maximum of each region. Create a max pooling layer using maxPooling2dLayer. A 2-D average pooling layer performs downsampling by dividing the input into rectangular pooling regions, then computing the average of each region. Create an average pooling layer using averagePooling2dLayer. Pooling layers follow the convolutional layers for down-sampling, hence, reducing the number of connections to the following layers. They do not perform any learning themselves, but reduce the number of parameters to be learned in the following layers. They also help reduce overfitting.

A max pooling layer returns the maximum values of rectangular regions of its input. The size of the rectangular regions is determined by the poolSize argument of maxPoolingLayer. For example, if poolSize is [2 3], then the layer returns the maximum value in regions of height 2 and width 3.

An average pooling layer outputs the average values of rectangular regions of its input. The size of the rectangular regions is determined by the poolSize argument of averagePoolingLayer. For example, if poolSize is [2 3], then the layer returns the average value of regions of height 2 and width 3.

Pooling layers scan through the input horizontally and vertically in step sizes you can specify using the 'Stride' name-value pair argument. If the pool size is smaller than or equal to the stride, then the pooling regions do not overlap.

For nonoverlapping regions (Pool Size and Stride are equal), if the input to the pooling layer is n-by-n, and the pooling region size is h-by-h, then the pooling layer down-samples the regions by h. That is, the output of a max or average pooling layer for one channel of a convolutional layer is n/h-by-n/h. For overlapping regions, the output of a pooling layer is (Input Size – Pool Size + 2\*Padding)/Stride + 1.

**Fully Connected Layer**

Create a fully connected layer using fullyConnectedLayer. A fully connected layer multiplies the input by a weight matrix and then adds a bias vector. The convolutional (and down-sampling) layers are followed by one or more fully connected layers. As the name suggests, all neurons in a fully connected layer connect to all the neurons in the previous layer. This layer combines all of the features (local information) learned by the previous layers across the image to identify the larger patterns. For classification problems, the last fully connected layer combines the features to classify the images. This is the reason that the output Size argument of the last fully connected layer of the network is equal to the number of classes of the data set. For regression problems, the output size must be equal to the number of response variables.

You can also adjust the learning rate and the regularization parameters for this layer using the related name-value pair arguments when creating the fully connected layer. If you choose not to adjust them, then trainNetwork uses the global training parameters defined by the training Options function. For details on global and layer training options.

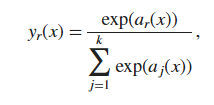
**Output Layers**

**Softmax and Classification Layers**

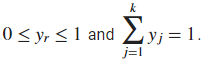
A softmax layer applies a softmax function to the input. Create a softmax layer using SoftMax Layer. A classification layer computes the cross-entropy loss for classification and weighted classification tasks with mutually exclusive classes. Create a classification layer using classificationLayer.

For classification problems, a softmax layer and then a classification layer usually follows the final fully connected layer.

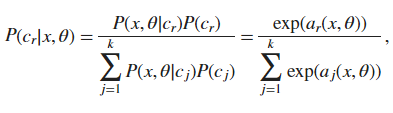
The output unit activation function is the softmax function:



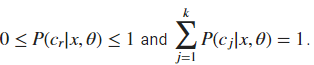
Were,

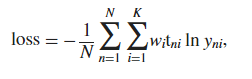


The softmax function is the output unit activation function after the last fully connected layer for multi-class classification problems:



Where, the conditional probability of the sample given class r, and P(cr) is the class prior probability.





where *N* is the number of samples, *K* is the number of classes, wi is the weight for class *i*, tni is the indicator that the *n*th sample belongs to the *i*th class, and yniis the output for sample *n* for class *i*, which in this case, is the value from the softmax function. In other words, yni is the probability that the network associates the *n*th input with class *i*.

**Disadvantages:**

* Using a CNN for fingerprint ID needs lots of computer time.
* Neural networks need lots of data for good training, making it tough for fingerprint datasets with various patterns.
* CNNs are like black boxes, hard to understand how they work.

**CHAPTER 4**

**PROPOSED METHOD**

Input Image

Pre-Processing

Minutiae features

Gabor Features

Fusion of Minutiae Features and Gabor features

Principal Component Analysis (PCA)

Create a feed-forward backpropagation network (BPNN)

Resize Image

Features

 Labels

Classification

* Arch
* Left Loop
* Right Loop
* Tented
* Whorl

Accuracy

Morphological Operations

* Dilate Image
* Erode Image
* Opening of Image

**Fig: Proposed Method Block Diagram**

The proposed methodology for fingerprint identification involves several sequential steps to enhance accuracy through the fusion of Gabor and Minutiae features using a Backpropagation Neural Network (BPNN) classifier. Initially, a dataset of fingerprint images is collected, followed by a preprocessing stage. This includes resizing images and applying morphological operations, such as dilation, erosion, and opening, to improve the quality and clarity of the features. Gabor features, capturing texture information, and minutiae extraction, identifying specific ridge characteristics, are then extracted. The fusion of Gabor and minutiae features aims to create a comprehensive representation of fingerprint patterns. To manage the high-dimensional feature space, Principal Component Analysis (PCA) is applied for dimensionality reduction. Subsequently, the reduced feature set along with their corresponding labels are loaded for the final classification step. A feed-forward BPNN is created to train the model to classify fingerprints into distinct categories, such as arches, left loops, right loops, tented arches, and whorls. The accuracy of the classification results is evaluated, demonstrating the effectiveness of the proposed methodology in accurately identifying and categorizing various fingerprint patterns.

**Pre-Processing:**

Preprocessing plays a crucial role in enhancing the quality of fingerprint images and facilitating accurate feature extraction. It begins with input fingerprint images, which may vary in size and quality. Resizing these images to a standardized dimension not only ensures consistency but also reduces computational complexity. Morphological operations are then applied to refine the fingerprint ridges and valleys, enhancing their clarity and continuity. Next, Gabor filters are utilized to extract both phase and texture features from the fingerprint image. Gabor phase features capture the orientation and direction of ridges, while Gabor texture features characterize the fine details and patterns within the ridges. These Gabor features provide a robust representation of the fingerprint's unique characteristics, enabling more accurate identification. By combining both phase and texture features, the preprocessing stage aims to create a comprehensive fingerprint representation that captures essential details while minimizing noise and irrelevant information. Overall, preprocessing lays the foundation for effective feature extraction and classification, ultimately improving the accuracy and reliability of fingerprint identification systems.

**Image Resize:**

Resizing an image in MATLAB involves altering its dimensions while preserving its aspect ratio or adjusting it as desired. The process is pivotal in various image processing tasks, including preprocessing for analysis or visualization. MATLAB offers versatile functions like "imresize" for resizing images conveniently. Upon calling this function, users specify the input image and the desired dimensions for the output image. MATLAB then employs interpolation techniques to resample pixel values, effectively adjusting the image's size. Bilinear or bicubic interpolation methods are commonly used to maintain image quality during resizing. Bilinear interpolation estimates new pixel values based on nearby pixels, while bicubic interpolation considers a larger neighbourhood for a smoother result. It's essential to note that resizing may lead to loss of information or aliasing if the new dimensions are significantly smaller than the original image. Conversely, enlarging an image can result in interpolation artifacts or blurring if the resolution is increased beyond the original. Thus, selecting appropriate interpolation methods and considering the specific requirements of the application are crucial for achieving desired results. In summary, MATLAB's resizing capabilities provide a flexible and efficient solution for adjusting image sizes while maintaining or enhancing image quality, supporting a wide range of image processing tasks.

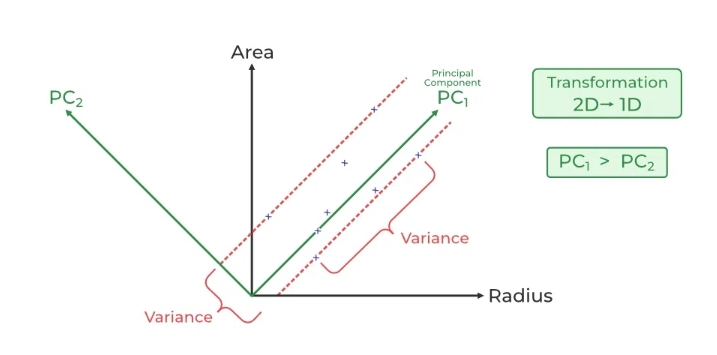
**Morphological Operations:**

Morphological operations in MATLAB are fundamental image processing techniques used to manipulate and analyze the structure of images based on their shape and form. These operations primarily involve the dilation and erosion of images, which are essential for tasks like noise removal, edge detection, and feature extraction. Dilation expands the boundaries of objects in an image, effectively thickening them, while erosion shrinks the object boundaries, making them thinner. These operations are often applied iteratively to enhance or refine specific features within an image. Additionally, MATLAB provides functions for other morphological operations such as opening, closing, and boundary extraction, which further aid in image enhancement and segmentation. Opening involves the sequential application of erosion followed by dilation and is commonly used for removing small objects and smoothing contours. Closing, on the other hand, performs dilation followed by erosion and is useful for filling small gaps and restoring object shapes. Boundary extraction isolates the outlines of objects in an image, facilitating subsequent analysis or feature extraction tasks. Overall, morphological operations in MATLAB offer powerful tools for preprocessing and analyzing images, contributing to various applications in fields such as medical imaging, computer vision, and remote sensing.

**Gabor phase and Gabor texture Features)**

Gabor phase and texture features are essential components in fingerprint analysis, particularly in MATLAB implementations. Gabor filters are mathematical functions used to extract features by convolving them with the input fingerprint image. Gabor phase features capture the orientation and direction of ridges in the fingerprint, providing valuable information about the overall structure. These features are computed by taking the phase component of the convolved image, which represents the local ridge orientation. Gabor texture features, on the other hand, characterize the fine details and patterns within the fingerprint ridges. This is achieved by considering the magnitude component of the convolved image, which highlights variations in the intensity or contrast of the ridges. In MATLAB, implementing Gabor phase and texture features typically involves defining parameters such as the frequency, orientation, and aspect ratio of the Gabor filters. These parameters influence the scale and sensitivity of the features extracted, allowing for customization based on the specific requirements of the fingerprint analysis task. Overall, leveraging MATLAB for Gabor feature extraction enables researchers and practitioners to efficiently process fingerprint images and extract discriminative features that are crucial for accurate identification and classification.

**PCA:**

****

As the number of features or dimensions in a dataset increases, the amount of data required to obtain a statistically significant result increase exponentially. This can lead to issues such as overfitting, increased computation time, and reduced accuracy of machine learning models this is known as the curse of dimensionality problems that arise while working with high-dimensional data.

As the number of dimensions increases, the number of possible combinations of features increases exponentially, which makes it computationally difficult to obtain a representative sample of the data and it becomes expensive to perform tasks such as clustering or classification because it becomes. Additionally, some [machine learning](https://www.geeksforgeeks.org/machine-learning/) algorithms can be sensitive to the number of dimensions, requiring more data to achieve the same level of accuracy as lower-dimensional data.

To address the [curse of dimensionality](https://www.geeksforgeeks.org/videos/curse-of-dimensionality-in-machine-learning/), [Feature engineering](https://www.geeksforgeeks.org/what-is-feature-engineering/)techniques are used which include feature selection and feature extraction. [Dimensionality reduction](https://www.geeksforgeeks.org/dimensionality-reduction/) is a type of feature extraction technique that aims to reduce the number of input features while retaining as much of the original information as possible.

What is Principal Component Analysis (PCA)?

[Principal Component Analysis](https://www.geeksforgeeks.org/principal-component-analysis-with-python/)(PCA) technique was introduced by the mathematician **Karl Pearson** in 1901**.** It works on the condition that while the data in a higher dimensional space is mapped to data in a lower dimension space, the variance of the data in the lower dimensional space should be maximum.

**Principal Component Analysis (PCA)**is a statistical procedure that uses an orthogonal transformation that converts a set of correlated variables to a set of uncorrelated variables. PCA is the most widely used tool in exploratory data analysis and in machine learning for predictive models. Moreover,

Principal Component Analysis (PCA) is an [unsupervised learning](https://www.geeksforgeeks.org/supervised-unsupervised-learning/) algorithm technique used to examine the interrelations among a set of variables. It is also known as a general factor analysis where regression determines a line of best fit.

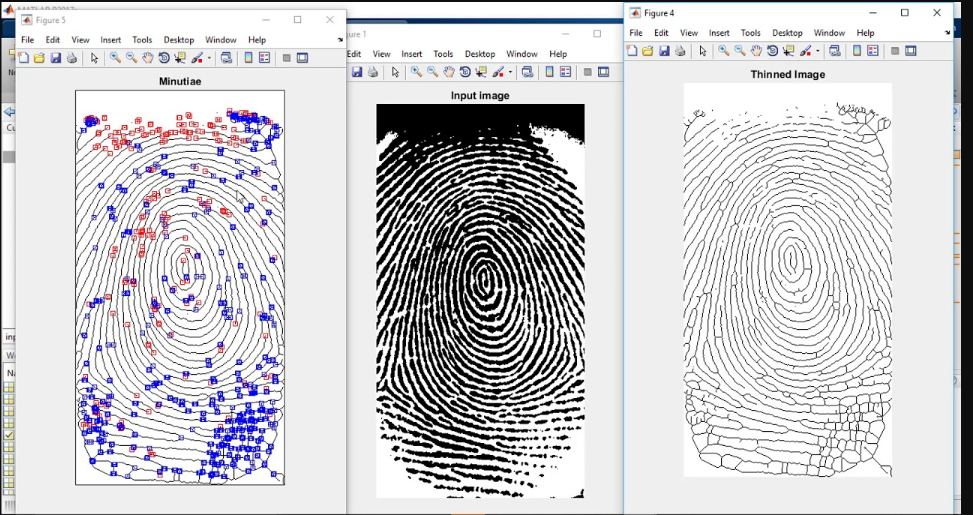
The main goal of Principal Component Analysis (PCA) is to reduce the dimensionality of a dataset while preserving the most important patterns or relationships between the variables without any prior knowledge of the target variables.

Principal Component Analysis (PCA) is used to reduce the dimensionality of a data set by finding a new set of variables, smaller than the original set of variables, retaining most of the sample’s information, and useful for the [regression and classification](https://www.geeksforgeeks.org/regression-classification-supervised-machine-learning/) of data.

* Principal Component Analysis (PCA) is a technique for dimensionality reduction that identifies a set of orthogonal axes, called principal components, that capture the maximum variance in the data. The principal components are linear combinations of the original variables in the dataset and are ordered in decreasing order of importance. The total variance captured by all the principal components is equal to the total variance in the original dataset.
* The first principal component captures the most variation in the data, but the second principal component captures the maximum [variance](https://www.geeksforgeeks.org/python-statistics-variance/) that is [orthogonal](https://www.geeksforgeeks.org/orthogonal-and-orthonormal-vectors-in-linear-algebra/) to the first principal component, and so on.
* Principal Component Analysis can be used for a variety of purposes, including data visualization, feature selection, and data compression. In data visualization, PCA can be used to plot high-dimensional data in two or three dimensions, making it easier to interpret. In feature selection, PCA can be used to identify the most important variables in a dataset. In data compression, PCA can be used to reduce the size of a dataset without losing important information.
* In Principal Component Analysis, it is assumed that the information is carried in the variance of the features, that is, the higher the variation in a feature, the more information that features carries.

**Minutiae features:**

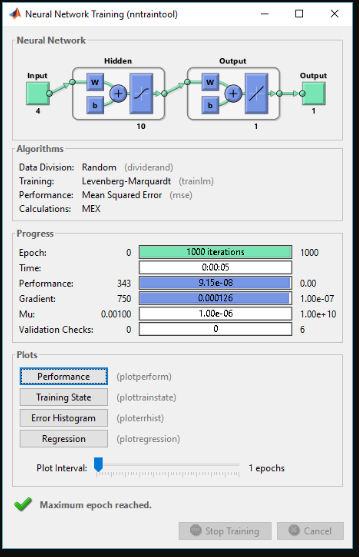
In MATLAB, Minutiae features refer to specific characteristics extracted from fingerprint images that are instrumental in fingerprint recognition and identification processes. These features typically include endpoints, where ridges terminate, and bifurcations, where ridges split into two branches. MATLAB provides various functions and algorithms to extract these minutiae features efficiently. The process usually involves preprocessing steps such as image enhancement and binarization to ensure clarity and consistency in the fingerprint image. Then, specialized algorithms are applied to detect minutiae points accurately. These algorithms often involve techniques like ridge thinning, skeletonization, and neighbourhood analysis to identify and characterize minutiae accurately. Once extracted, these minutiae features serve as distinctive landmarks, forming a unique fingerprint representation. MATLAB offers tools for feature extraction, visualization, and analysis, enabling researchers and practitioners to manipulate and study minutiae features effectively. Additionally, MATLAB's extensive libraries and functions support further processing steps like matching, classification, and pattern recognition, facilitating comprehensive fingerprint analysis. Overall, the utilization of MATLAB for minutiae feature extraction provides a robust and versatile framework for fingerprint biometrics research and application development.



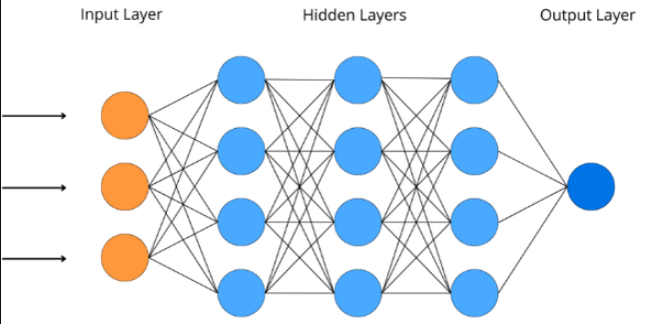
The two most prominent local ridge characteristics are: 1) ridge ending and, 2) ridge bifurcation. A ridge ending is defined as the point where a ridge ends abruptly. A ridge bifurcation is defined as the point where a ridge forks or diverges into branch ridges. Collectively, these features are called minutiae.

**Create a feed-forward backpropagation network:**

In MATLAB, creating a feed-forward backpropagation neural network involves several steps to build a robust and efficient model for various applications. First, define the architecture of the network by specifying the number of input nodes, hidden layers, and output nodes. Using MATLAB's Neural Network Toolbox functions, such as `feedforward net`, you can easily create the initial structure of the network. Next, customize the network by adjusting parameters like the number of neurons in each layer, activation functions, and learning algorithms. MATLAB provides a range of options for activation functions such as sigmoid, tanh, and ReLU, as well as various optimization algorithms like stochastic gradient descent (SGD) and Levenberg-Marquardt. Once the network structure is defined, data preparation is crucial. Ensure that your input data is properly formatted and normalized to improve convergence and performance. Then, use MATLAB's built-in functions like `train` to train the network using your dataset. During training, the backpropagation algorithm updates the weights and biases of the network to minimize the error between predicted and actual outputs. Finally, evaluate the trained network using validation or test datasets to assess its performance metrics such as accuracy, loss, and convergence. Through MATLAB's intuitive interface and powerful toolbox, creating a feed-forward backpropagation neural network becomes a straightforward process, enabling users to develop sophisticated models for various tasks efficiently.



In MATLAB, the "sim" function is a pivotal component for evaluating the performance of a neural network model. Primarily, it serves the purpose of simulating the neural network's response to input data. Upon providing input patterns to the network, "sim" computes the corresponding output responses based on the network's current configuration, including the weights and biases learned during training. This function essentially mimics the forward propagation process within the neural network, where inputs are transformed through successive layers of interconnected neurons to produce an output. The "sim" function thus enables users to assess the accuracy and effectiveness of their trained neural network model by comparing the predicted outputs with the actual target outputs for the given input data. Additionally, "sim" allows for batch processing of multiple input patterns, enhancing efficiency in evaluating the network's performance across a range of scenarios. Overall, the "sim" function in MATLAB plays a crucial role in the iterative process of developing and refining neural network models, providing valuable insights into their capabilities and guiding adjustments to improve their predictive accuracy and generalization abilities.



**Accuracy:**

In MATLAB, "Accuracy" is a common performance metric used to evaluate the overall effectiveness of a classification model, particularly in binary or multi-class classification tasks. It measures the proportion of correctly classified samples over the total number of samples in the dataset. To understand Accuracy, it's essential to know the concepts of True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN):

* True Positives (TP): The number of samples correctly classified as positive by the classifier.
* False Positives (FP): The number of negative samples that were incorrectly classified as positive by the classifier.
* True Negatives (TN): The number of samples correctly classified as negative by the classifier.
* False Negatives (FN): The number of positive samples that were incorrectly classified as negative by the classifier.

Accuracy is calculated using the following formula:

Accuracy = (TP + TN) / (TP + TN + FP + FN)

In other words, Accuracy is the ratio of the correctly classified samples (both positive and negative) to the total number of samples in the dataset.

In MATLAB, confusion matrices are widely used for evaluating the performance of classification models, particularly in machine learning and pattern recognition tasks. A confusion matrix provides a comprehensive summary of the predicted versus actual classifications made by a model across different classes. From the confusion matrix, several performance metrics can be derived, including sensitivity, specificity, precision, and F-measure. Sensitivity, also known as the true positive rate, measures the proportion of actual positive cases that were correctly identified by the model. Specificity, on the other hand, quantifies the ability of the model to correctly identify negative cases. Precision represents the proportion of true positive predictions among all positive predictions made by the model. It indicates the accuracy of the positive predictions made by the model. The F-measure, which is the harmonic mean of precision and recall (sensitivity), provides a balanced measure of a model's performance across both positive and negative classes. These metrics are crucial for assessing the effectiveness and reliability of classification models, helping researchers and practitioners to make informed decisions about the model's suitability for specific applications. MATLAB provides built-in functions for computing confusion matrices and deriving these performance metrics, enabling efficient evaluation and optimization of classification algorithms.

**CHAPTER 5**

**ADVANTAGES AND APPLICATIONS**

**Advantages:**

* Combining Gabor and Minutiae features boosts fingerprint system accuracy, securing access.
* BPPN learns well, helps classify, makes fingerprint ID smoother, more effective.
* System uses BPPN and Minutiae, works well for all fingerprint types.
* Minutiae features enable unique and precise fingerprint identification.

**Applications:**

* Biometric Security.
* Fingerprint help police catch criminals and track suspects in investigations.
* Fingerprint systems help border control verify travelers and spot fake documents.
* Fingerprint security safeguards banking, blocks unauthorized access to accounts and data.

**CHAPTER 6**

**RESULTS**

****

**Fig: Input Image**

****

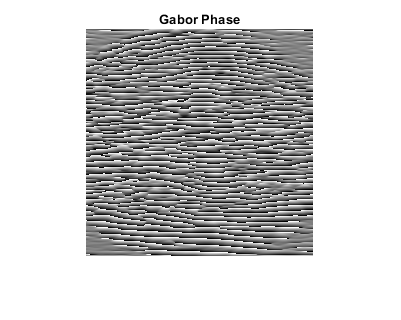
**Fig: Resize Image**

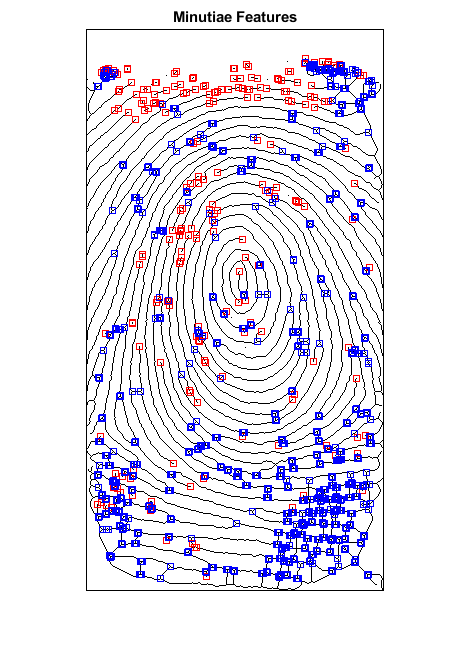
****

**Fig: Dilated Image**

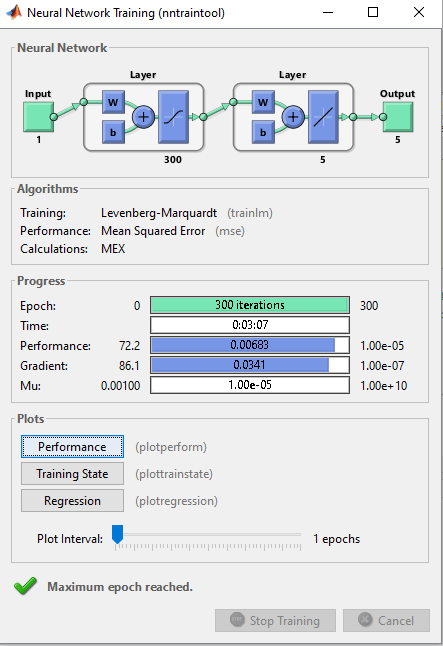
****

**Fig: Erode Image**

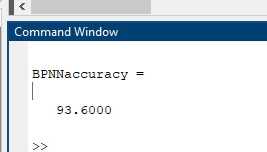
****

**Fig: Gabor Phase Image**

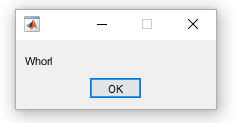
**Fig: Minutiae Features**

****

**Fig: Training Process**

****

**Fig: Accuracy**

****

**Fig: Final Output**

|  |  |  |
| --- | --- | --- |
| **S. No** | **Existing Method** | **Proposed Method** |
| 1 | 78 | 86.4 |
| 2 | 78 | 93.6 |
| 3 | 78 | 90 |
| 4 | 78 | 93.6 |
| 5 | 78 | 90 |

**Fig: Difference Between Existing Method and Proposed Method**

**Fig: Comparison Graph**

**CHAPTER 7**

**CONCLUSION**

In conclusion, the fusion of Gabor and Minutiae features coupled with a BPNN classifier presents a robust approach to fingerprint identification. Through the systematic process involving preprocessing, feature extraction, fusion, dimensionality reduction, and classification, the system achieves accurate recognition of various fingerprint patterns including arches, loops, tented arches, and whorls. The application of morphological operations, PCA for dimensionality reduction, and the utilization of a BPNN classifier contribute to the system's reliability and efficiency. This methodology demonstrates promising results in enhancing the accuracy of biometric authentication systems, thereby ensuring secure access control. Moving forward, further research could explore optimizations to streamline the process and potentially integrate additional features or algorithms to enhance the system's performance in real-world applications.

**REFERENCES**

[1] Hasan H, Abdul-Kareem S. Fingerprint image enhancement and recognition algorithms: a survey. Neural Comput Appl 2013;(23):1605–10.

[2] Win KN, Li K, Chen J, Viger PF, Li K. Fingerprint classification and identification algorithms for criminal investigation: a survey. Fut Gener Comput Syst 2020; 110:758–71.

[3] Nazarkevych M, Riznyk O, Samotyy V, Dzelendzyak U. Detection of regularities in the parameters of the Ateb-Gabor method for biometric image filtration. East Eur J Enterp Technol 2019;1(2–97):57–65.

[4] Shemmary ENAAl. Classification of fingerprint images using neural networks technique. J Eng (JOE) 2012;1(3):40–8.

[5] Leung KC, Leung CH. Improvement of fingerprint retrieval by a statistical classifier. IEEE Trans Inform Forens Secur 2011;6(1):59–69.

[6] Wan GC, Li MM, Xu H, Kang WH, Rui JW, Tong MS. XFinger-net: pixel-wise segmentation method for partially defective fingerprint based on attention gates and U-net. Sensors 2020;20:4473.

[7] AlShehri H, Hussain M, AboAlSamh H, AlZuair M. A large-scale study of fingerprint matching systems for sensor interoperability problem. Sensors 2018;18: 1008.

[8] Wu F, Zhu J, Guo X. Fingerprint pattern identification and classification approach based on convolutional neural networks. Neural Comput Appl 2020;32: 5725–34.

[9] Takahashi, A., Koda, Y., Ito, K., & Aoki, T. (2020). Fingerprint feature extraction by combining texture, minutiae, and frequency spectrum using multi-task CNN, 2020, arXiv preprint arXiv:2008.11917.

[10] Borra SR, Reddy GJ, Reddy ES. Classification of fingerprint images with the aid of morphological operation and AGNN classifier. Appl Comput Inform 2018;14: 166–76.

[11] Peralta D, Triguero I, García S, Saeys Y, Benitez JM, Herrera F. On the use of convolutional neural networks for robust classification of multiple fingerprint captures. Int J Intell Syst 2018;33:213–30.

[12] Peralta D, Galar M, Triguero I, Paternain D, García S, Barrenechea E, Beníteza JM, Bustince H, Herrera F. A survey on fingerprint minutiae-based local matching for verification and identification: taxonomy and experimental evaluation. Expert Syst Appl 2017;81:251–67.

[13] Jain AK, Feng J, Nandakumar K. Fingerprint matching. IEEE Comput 2010;43:36–44.

[14] Ge W, Sun M, Wang X. An incremental two-dimensional principal component analysis for object recognition. Math Probl Eng 2018:1–13. 2018, 2018.

[15] Fiumara G., Flanagan P., Grantham J., Bandini B., Ko K., Libert J., NIST special database 300 uncompressed plain and rolled images from fingerprint cards, national institute of standards and technology, 2018 <https://doi.org/10.18434/T4/1502472>.

[16] Maio D, Maltoni D, Cappelli R, Wayman JL, Jain AK. FVC2000: Fingerprint verification competition. IEEE Trans. Pattern Anal. Mach. Intell 2002;24:402–12.

[17] Maio D, Maltoni D, Cappelli R, Wayman JL, Jain AK. FVC2002: second fingerprint verification competition. Proc Int Conf Pattern Recognit 2002;14:4–20.

[18] Maio D, Maltoni D, Cappelli R, Wayman JL, Jain AK. FVC2004: Third fingerprint verification competition. Proc. Int. Conf. Biom. Authent 2004;3072:1–7.

[19] Yang, W., Wang S., Hu, J., Zheng G., Valli C., Security and accuracy of fingerprint-based biometrics: a review, symmetry, 2019, 11, 141.

[20] Shereena VB, Julie MD. Significance of dimensionality reduction in image processing. Signal Image Process, Int J (SIPIJ) 2015;6:27–42.

[21] Rajanna U, Erol A, Bebis G. A comparative study on feature extraction for fingerprint classification and performance improvements using rank-level fusion. Pattern Anal Appl 2010;13(3):263–72.

[22] Le T, Van HT. Fingerprint reference point detection for image retrieval based on symmetry and variation. Pattern Recogn 2012;45(9):3360–72.

[23] Nguyen HT. Fingerprints classification through image analysis and machine learning method. Algorithms 2019;12(11):241.

[24] Wu JD, Liu CT. Finger-vein pattern identification using SVM and neural network technique. Expert Syst Appl 2011;38(11):14284–9.

[25] Militello C, Rundo L, Vitabile S, Conti V. Fingerprint classification based on deep learning approaches: experimental findings and comparisons. Symmetry. 2021; 13(5):750. <https://doi.org/10.3390/sym13050750>.

[26] Yang, J., Wu, Z. and Zhang J., A robust fingerprint identification method by deep learning with Gabor filter multidimensional feature expansion; 4th international conference, icccs 2018, Haikou, China, June 8–10, 2018, pp. 447–57.

[27] Michelsanti D., Ene, A.D., Guichi, Y., Stef, R., Nasrollahi, K. and Moeslund, T.B., Fast fingerprint classification with deep neural networks, Visual Analysis of People (VAP) Laboratory, Aalborg University, Aalborg, Denmark, 2017. ISBN: 978-989-758-226-4, DOI: 10.5220/0006116502020209.

[28] Everingham M, Van Gool L, Williams C. The pascal visual object classes (voc) challenge. Int J Comput Vis 2010;88:303–38.

[29] Hong JH, Min JK, Cho UK. Fingerprint classification using one-vs-all support vector machines dynamically ordered with Bayes classifiers. Pattern Recognit 2008;41:662–71.

[30] Strobl C. An introduction to recursive partitioning: rationale, application and characteristics of classification and regression trees, bagging and random forests. Psychol Methods 2009;14:323–48.

[31] Cristianini N, Shawe-Taylor J. An introduction to support vector machines (and other kernel-based learning methods). Cambridge, UK: Cambridge University Press; 2000. p. 190.

[32] Mohamed SM, Nyongesa HO. Automatic fingerprint classification system using fuzzy neural techniques, Fuzzy Systems. In: Proceedings of the 2002 IEEE World Congress on Computational Intelligence, 2002 IEEE International Conference on Fuzzy Systems, Honolulu, HI, USA, 12–17. 1; May 2002. p. 358–62.

[33] Sun Y, Abeywickrama S, Jayasinghe L, Yuen C, Chen J, Zhang M. Micro-Doppler signature-based detection, classification, and localization of small UAV with long short-term memory neural network. IEEE Trans Geosci Rem Sens 2020.

[34] Wu F, Zhu J, Guo X. Fingerprint pattern identification and classification approach based on convolutional neural networks. Neural Comput Appl 2019;32: 5725–34. 2020.

[35] Zhong Z, Jin L, Xie Z. High performance offline handwritten chinese character recognition using googlenet and directional feature maps. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR). IEEE; 2015. p. 846–50.

[36] Aung Swe Swe, Nagayama Itaru, Tamaki Shiro. Adaptive morphological operation for high-performance weather image processing. IEIE Trans Smart Process Comput 2018;7(6):424–32.

[37] Bae Jungyoon, Choi Han-Soo, Kim Sujin, Kang Myungjoo. Fingerprint image denoising and inpainting using convolutional neural network. J Korean Soc Ind Appl Math 2020;24(4):363–74.

[38] Biswas R, Vasan A, Roy SS. Dilated deep neural network for segmentation of retinal blood vessels in fundus images. Iran J Sci Technol Trans Electr Eng 2020;44: 505–18. https://doi.org/10.1007/s40998-019-00213-7.

**BIBLIOGRAPHY**

**Introduction To MATLAB**

What Is MATLAB?

The name MATLAB stands for Matrix Laboratory. The software is built up around vectors and matrices. This makes the software particularly useful for linear algebra but MATLAB is also a great tool for solving algebraic and differential equations and for numerical integration. MATLAB has powerful graphic tools and can produce nice pictures in both 2D and 3D. It is also a programming language, and is one of the easiest programming languages for writing mathematical programs. These factors make MATLAB an excellent tool for teaching and research.

MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects. It integrates computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment: it has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems.

MATLAB abilities a family of add-on software program utility software application software program software utility software-unique solutions called toolboxes. Very essential to maximum customers of MATLAB, toolboxes assist you to studies and observe specialized technology. Toolboxes are entire collections of MATLAB abilities (M-files) that increase the MATLAB surroundings to remedy precise schooling of problems. Areas in which toolboxes are to be had embody signal processing, manipulate systems, neural networks, fuzzy correct judgment, wavelets, simulation, and hundreds of others.

It has powerful built-in routines that enable a very wide variety of computations. It also has easy to use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. The software package has been commercially available since 1984 and is now considered as a standard tool at most universities and industries worldwide.

**Brief History of MATLAB:**

Cleve Moler, the chairman of the computer science department at the University of New Mexico, started developing MATLAB in the late 1970s. The first MATLAB® was not a programming language; it was a simple interactive matrix calculator. There were no programs, no toolboxes, no graphics and no ODEs or FFTs. He designed it to give his student’s access to LINPACK and EISPACK without them having to learn FORTRAN. It soon spread to other universities and found a strong audience within the applied mathematics community. The mathematical basis for the first version of MATLAB was a series of research papers by J. H. Wilkinson and 18 of his colleagues, published between 1965 and 1970 and later collected in Handbook for Automatic Computation, Volume II, Linear Algebra*,* edited by Wilkinson and C. Reinsch. These papers present algorithms, implemented in Algol 60, for solving matrix linear equation and Eigen value problems.

In the 1970s and early 1980s, I was teaching Linear Algebra and Numerical Analysis at the University of New Mexico and wanted my students to have easy access to LINPACK and EISPACK without writing FORTRAN programs. By “easy access,” I meant not going through the remote batch processing and the repeated edit-compile-link-load-execute process that was ordinarily required on the campus central mainframe computer. Jack little, an engineer, was exposed to it during a visit Moler made to Stanford University in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in C and founded Math Works in 1984 to continue its development. These rewritten libraries were known as JACKPAC. In 2000, MATLAB was rewritten to use a newer set of libraries for matrix manipulation, LAPACK. MATLAB was first adopted by researchers and practitioners in control engineering, little’s specialty, but quickly spread to many other domains. It is now also used in education, in particular the teaching of linear algebra and numerical analysis, and is popular amongst scientists involved in video processing**.**

## **EISPACK and LINPACK**:

In 1970, a group of researchers at Argonne National Laboratory proposed to the U.S. National Science Foundation (NSF) to “explore the methodology, costs, and resources required to produce, test, and disseminate high-quality mathematical software and to test, certify, disseminate, and support packages of mathematical software in certain problem areas.” The group developed EISPACK (Matrix Eigen system Package) by translating the Algol procedures for Eigen value problems in the handbook into FORTRAN and working extensively on testing and portability. The first version of EISPACK was released in 1971 and the second in 1976.

In 1975, four of us Jack Dongarra, Pete Stewart, Jim Bunch, and myself proposed to the NSF another research project that would investigate methods for the development of mathematical software. A byproduct would be the software itself, dubbed LINPACK, for Linear Equation Package. This project was also centered at Argonne. LINPACK originated in FORTRAN; it did not involve translation from Algol. The package contained 44 subroutines in each of four numeric precisions. In a sense, the LINPACK and EISPACK projects were failures. We had proposed research projects to the NSF to “explore the methodology, costs, and resources required to produce, test, and disseminate high-quality mathematical software.” We never wrote a report or paper addressing those objectives. We only produced software.

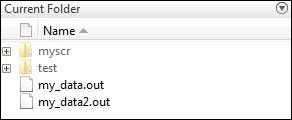
So, I studied Niklaus Wirth’s book Algorithms + Data Structures *=* Programs and learned how to parse programming languages. I wrote the first MATLAB an acronym for Matrix Laboratory in FORTRAN, with matrix as the only data type. The project was a kind of hobby, a new aspect of programming for me to learn and something for my students to use. There was never any formal outside support, and certainly no business plan. This first MATLAB was just an interactive matrix calculator. This snapshot of the start-up screen shows all the reserved words and functions. There are only 71. To add another function, you had to get the source code from me, write a FORTRAN subroutine, add your function name to the parse table, and recompile MATLAB.

**Starting MATLAB:**

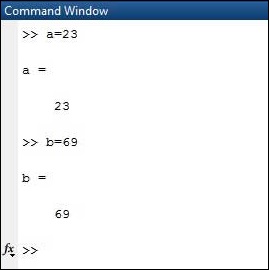
After logging into your account, you can enter MATLAB by double-clicking on the MATLAB shortcut icon (MATLAB 7.0.4) on your Windows desktop. When you start MATLAB, a special window called the MATLAB desktop appears. The desktop is a window that contains other windows. The major tools within or accessible from the desktop are:

* The Command Window
* The Command History
* The Workspace
* The Current Directory
* The Help Browser

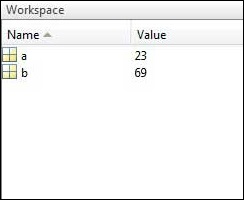
**Current Folder:** This panel allows you to access the project folders and files.



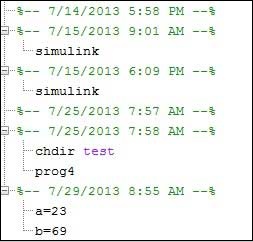
**Command Window:** This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).



**Workspace:**  The workspace shows all the variables created and/or imported from files.



**Command History:** This panel shows or return commands that are entered at the command line.



**Help Browser:**

The critical way to get assist online is to use the MATLAB help browser, opened as a separate window every through clicking at the question mark photograph (?) on the computing tool toolbar, or through manner of typing assist browser on the spark off in the command window. The assist Browser is an internet browser blanketed into the MATLAB computing tool that shows a Hypertext Markup Language (HTML) file. The Help Browser consists of panes, the help navigator pane, used to find out information, and the show pane, used to view the information. Self-explanatory tabs apart from navigator pane are used to performs are searching out.

**MATLAB language:**

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.

**MATLAB working environment:**

This is the set of tools and facilities that you work with as the MATLAB user or programmer. It includes facilities for managing the variables in your workspace and importing and exporting data. It also includes tools for developing, managing, debugging, and profiling M-files, MATLAB's applications.

**MATLAB mathematical function library:**

This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.

**MATLAB Application Program Interface (API):**

This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

**MATLAB DESKTOP:**

MATLAB Desktop is the precept MATLAB utility window. The computing tool includes five sub home windows, the command window, the workspace browser, the modern-day-day list window, the command records window, and one or greater decide domestic windows, which is probably confirmed high-quality on the identical time due to the truth the client suggests a photo. The command window is in which the character types MATLAB instructions and expressions at the spark off (>>) and in which the output of these commands is displayed. MATLAB defines the workspace because the set of variables that the client creates in a bit consultation. The workspace browser suggests those variables and some facts about them. Double clicking on a variable within the workspace browser launches the Array Editor, which may be used to gain statistics and profits instances edit exceptional homes of the variable.

The modern-day-day-day Directory tab above the workspace tab suggests the contents of the cutting-edge list, whose path is shown inside the modern-day list window. For example, in the home windows on foot machine the path is probably as follows: C: MATLAB Work, indicating that listing “artwork” is a subdirectory of the number one list “MATLAB”; WHICH IS INSTALLED IN DRIVE C.

Clicking on the arrow within the modern list window suggests a listing of these days used paths. Clicking at the button to the right of the window permits the individual to trade the present-day listing. MATLAB uses a seeking out path to find out M-documents and one-of-a-type MATLAB associated documents, which can be put together in directories within the computer document tool. Any report run in MATLAB need to be dwelling in the modern-day-day listing or in a list that is on is looking for course. By default, the documents supplied with MATLAB and math works toolboxes are included inside the searching out direction. The first-rate manner to look which directories are on the searching out route. The satisfactory manner to appearance which directories are speedy the quest route, or to characteristic or regulate a searching for course, is to pick out outset path from the File menu the computing device, and then use the set course talk discipline. It is proper exercise to feature any generally used directories to the hunt route to avoid again and again having the exchange the cutting-edge-day listing.

The Command History Window contains a file of the instructions a person has entered in the command window, together with every contemporary-day and former MATLAB periods. Previously entered MATLAB instructions can be determined on and re-completed from the command statistics window thru proper clicking on a command or series of commands. This movement launches a menu from which to select numerous options similarly to executing the commands. This is useful to select out abilities options in addition to executing the instructions. This is a beneficial feature at the equal time as experimenting with numerous commands in a piece session.

**Using the MATLAB Editor to create M-Files:**

The MATLAB editorial manager is a literary substance proof-reader particular for growing M-facts and a graphical MATLAB debugger. The supervisor can seem in a window through command facts technique for itself, or it is probably a right-clicking inside the PC. M-information this gadget signified through the use of the expansion .M, as in pixel up.M. The MATLAB editorial supervisor window has a few draws down menus for obligations collectively with sparing, seeing, and troubleshooting facts. Since it plays more than one easy test and furthermore affects utilization of shade to separate among exclusive variables of code, this article editorial supervisor is often supported due to reality the system of a need for composing and altering M-talents. To open the manager, type at enact opens the M-document filename. M in a supervisor window, sorted out for enhancing. As stated earlier than, the file should be inside the cutting-edge posting, or in a posting in the seeking out direction.

## **Features of MATLAB:**

Following are the basic features of MATLAB.

* It is a high-level language for numerical computation, visualization and application development.
* It also provides an interactive environment for iterative exploration, design and problem solving.
* It provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.
* It provides built-in graphics for visualizing data and tools for creating custom plots.
* MATLAB's programming interface gives development tools for improving code quality maintainability and maximizing performance.
* It provides tools for building applications with custom graphical interfaces.
* It provides functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET and Microsoft Excel.

## **Uses of MATLAB:**

MATLAB is widely used as a computational tool in science and engineering encompassing the fields of physics, chemistry, math and all engineering streams. It is used in a range of applications including

* Signal Processing and Communications
* Video and Video Processing
* Control Systems
* Test and Measurement
* Computational Finance
* Computational Biology

**Applications of MATLAB:**

MATLAB can be used as a tool for simulating various electrical networks but the recent developments in MATLAB make it a very competitive tool for Artificial Intelligence, Robotics, Video processing, Wireless communication, Machine learning, Data analytics and whatnot. Though it’s mostly used by circuit branches and mechanical in the engineering domain to solve a basic set of problems its application is vast. It is a tool that enables computation, programming and graphically visualizing the results. The basic data element of MATLAB as the name suggests is the Matrix or an array. MATLAB toolboxes are professionally built and enable you to turn your imaginations into reality. MATLAB programming is quite similar to C programming and just requires a little brush up of your basic programming skills to start working with.

Below are a few applications of MATLAB –

* **Statistics and machine learning (ML)**

This toolbox in MATLAB can be very handy for the programmers. Statistical methods such as descriptive or inferential can be easily implemented. So is the case with machine learning. Various models can be employed to solve modern-day problems. The algorithms used can also be used for big data applications.

* **Curve fitting**

The curve fitting toolbox helps to analyze the pattern of occurrence of data. After a particular trend which can be a curve or surface is obtained, its future trends can be predicted. Further plotting, calculating integrals, derivatives, interpolation, etc. can be done.

* **Control systems**

Systems nature can be obtained. Factors such as closed-loop, open-loop, its controllability and observability, Bode plot, NY Quist plot, etc. can be obtained. Various controlling techniques such as PD, PI and PID can be visualized. Analysis can be done in the time domain or frequency domain.

* **Signal Processing**

Signals and systems and digital signal processing are taught in various engineering streams. But MATLAB provides the opportunity for proper visualization of this. Various transforms such as Laplace, Z, etc. can be done on any given signal. Theorems can be validated. Analysis can be done in the time domain or frequency domain. There are multiple built-in functions that can be used.

* **Mapping**  
  Mapping has multiple applications in various domains. For example, in Big Data, the Map Reduce tool is quite important which has multiple applications in the real world. Theft analysis or financial fraud detection, regression models, contingency analysis, predicting techniques in social media, data monitoring, etc. can be done by data mapping.
* **Deep learning**

It’s a subclass of machine learning which can be used for speech recognition, financial fraud detection, and medical video analysis. Tools such as time-series, Artificial neural network (ANN), Fuzzy logic or combination of such tools can be employed.

* **Financial analysis**

An entrepreneur before starting any endeavor needs to do a proper survey and the financial analysis in order to plan the course of action. The tools needed for this are all available in MATLAB. Elements such as profitability, solvency, liquidity, and stability can be identified. Business valuation, capital budgeting, cost of capital, etc. can be evaluated.

* **Video processing**

The most common application that we observe almost every day are bar code scanners, selfie (face beauty, blurring the background, face detection), video enhancement, etc. The digital video processing also plays quite an important role in transmitting data from far off satellites and receiving and decoding it in the same way. Algorithms to support all such applications are available.

* **Text analysis**

Based on the text, sentiment analysis can be done. Google gives millions of search results for any text entered within a few milliseconds. All this is possible because of text analysis. Handwriting comparison in forensics can be done. No limit to the application and just one software which can do this all.

* **Electric vehicles designing**

Used for modelling electric vehicles and analyse their performance with a change in system inputs. Speed torque comparison, designing and simulating of a vehicle, whatnot.

* **Aerospace**

This toolbox in MATLAB is used for analysing the navigation and to visualize flight simulator.

* **Audio toolbox**

Provides tools for audio processing, speech analysis, and acoustic measurement. It also provides algorithms for audio and speech feature extraction and audio signal transformation.

**DIGITAL IMAGE/VIDEO PROCESSING**

**Digital image processing:**

Digital Image Processing means processing digital image by means of a digital computer. We can also say that it is a use of computer algorithms, in order to get enhanced image either to extract some useful information.

**Image:**

An image is defined as a two-dimensional function, F (x, y), where x and y are spatial coordinates, and the amplitude of F at any pair of coordinates (x, y) is called the intensity of that image at that point. When x, y, and amplitude values of F are finite, we call it a digital image. In other words, an image can be defined by a two-dimensional array specifically arranged in rows and columns. Image is composed of a finite number of elements, each of which elements have a particular value at a particular location. These elements are referred to as picture elements, image elements, and pixels.

A Pixel is most widely used to denote the elements of an Image. To be processed digitally, it has to be **sampled** and transformed into a matrix of numbers. Since a computer represents the numbers using finite precision, these numbers have to be **quantized**to be represented digitally. Digital image processing consists of the manipulation of those finite precision numbers. The processing of digital images can be divided into several classes: **image enhancement, image restoration, image analysis,** and image compression.

## How a digital image is formed?

Since capturing an image from a camera is a physical process. The sunlight is used as a source of energy. A sensor array is used for the acquisition of the image. So, when the sunlight falls upon the object, then the amount of light reflected by that object is sensed by the sensors, and a continuous voltage signal is generated by the amount of sensed data. In order to create a digital image, we need to convert this data into a digital form. This involves sampling and quantization. (They are discussed later on). The result of sampling and quantization results in a two-dimensional array or matrix of numbers which are nothing but a digital image.

**Image processing mainly include the following steps:**

* Importing the image via image acquisition tools.
* Analyzing and manipulating the image.
* Output in which result can be altered image or a report which is based on analyzingthat image.

An image can be portrayed as a - dimensional trademark f (x, y), in which x and y are spatial directions, and the sufficiency of any combine of instructions (x, y) is known as the pressure or darkish degree of the image at that inconvenience. Whenever x, y and the abundance estimations off are on the entire confined discrete quantities, we call the picture a virtual photo. The district of DIP alludes to getting ready computerized photo through strategies for to method for MATLAB. Manner of the use of advanced pc. Computerized image incorporates of a confined form of things, every one in every of which has a chosen location and fee. The components are alluded to as pixels.

Vision is the maximum innovative of our sensor, so it isn't sudden that photograph play the unmarried greatest important component in human conviction. Nonetheless, in appraisal to humans, who are controlled to the visible band of the EM variety imaging machines cover nearly the complete EM range, starting from gamma to radio waves. They can highlight also on previews created with the valuable useful manual of benefits that people aren't conscious of accomplice with airship picture. There isn't commonly any present settlement among creators concerning in which photo managing stops and specific associated districts nearby aspect photo evaluation& workstation imaginative and prescient start.

In a few instances a difference is made through the use of characterizing picture handling as an area wherein each the information and yield at a way are snap shots. This is constraining and predominantly manufactured restriction. The district of image investigation (photograph getting to know) is in amongst photograph getting ready and PC imaginative and insightful. There aren't any easy restrictions in the continuum from picture preparing at one prevent to complete ingenious and sensible on the inverse. In any case, one precious worldview is to revel in as a primary challenge three types of automatic procedures in this continuum: low-, mid-, and radical affirmation methodologies. Low-certificate way includes crude obligations which incorporates image preparing to reduce clamor, appraisal improvement and photo cleansing. A low-certificates approach is described through the way that very it inputs and yields are previews.

## Mid-degree method on photographs incorporates of commitments which exemplify division, depiction of that question decreases them to a form becoming for pc getting ready and category of man or woman devices. A mid-degree method is portrayed thru technique for the reality that its resources of info broadly speaking are images anyway its yields are residences extricated from the one’s photographs. At lengthy remaining better-degree making ready carries "Making history" of a meeting of analyzed gadgets, as in photo assessment and at a broadened path give up of the continuum performing the psychological abilities often connected with human imaginative and prescient. Computerized image making ready, as formally depicted is utilized accurately in a huge kind of locales of astonishing social and financial price.

**PHASES OF IMAGE PROCESSING:**

* **Acquisition:**

It could be as simple as being given an image which is in digital form. The main work involves: a) Scaling b) Color conversion (RGB to Gray or vice-versa).

* **Image enhancement:**

It is amongst the simplest and most appealing in areas of Image Processing it is also used to extract some hidden details from an image and is subjective.

* **Image restoration:** It also deals with appealing of an image but it is objective (Restoration is based on mathematical or probabilistic model or image degradation).
* **Color image processing:**

It deals with pseudo color and full color image processing color models are applicable to digital image processing.

* **Wavelets and Multi-resolution processing:**

It is foundation of representing images in various degrees.

* **Image compression:**

It involves in developing some functions to perform this operation. It mainly deals with image size or resolution.

* **Morphological processing:**

It deals with tools for extracting image components that are useful in the representation & description of shape.

* **Segmentation:**

It includes partitioning an image into its constituent parts or objects. Autonomous segmentation is the most difficult task in Image Processing.

* **Representation and description:**

It follows output of segmentation stage, choosing a representation is only the part of solution for transforming raw data into processed data.

* **Object detection and recognition:**

It is a process that assigns a label to an object based on its descriptor.

## **dip.PNG**

**Color image:**

It may be spoken to with the aid of techniques for manner of three capacities, R (xylem) for purple, G (xylem) for inexperienced and B (xylem)for blue. An image may be nonstop with renowned to the x and y arranges and moreover in sufficiency. Changing over this type of picture to digital form requires that the guidelines in addition to the adequacy to be digitized. Digitizing the set up's qualities is called analysing. Digitizing the abundancy esteems is known as quantization.

**Grayscale image:** ƒ

The image has 8 bits and 256 tons of grey; 1 = black and 255 = white. ƒ Requires 8 times more saving space than a line-art image. Suitable for presenting black and white photographs, for instance. Can be used in printing office.

**Image Types:**

The tool compartment underpins 4 types of images:

1. Intensity of pixels;

2. Twofold images;

3. Filed images;

4. R G B images.

Most monochrome image making ready sports are finished utilizing parallel or force fix, so our underlying highlight is on this image composes. Filed and RGB shading images.

**Intensity Images:**

A profundity picture is a measurement lattice whose traits were scaled to talk to goals. At the point while the components of a profundity photo are of class unit8, or elegance unit sixteen, they have complete quantity traits in the collection [0,255] and [0, 65535], for my part. On the off danger that the picture is of class twofold, the qualities are skimming phase numbers. Estimations of scaled, twofold pressure images are within the assortment [0, 1] by means of methods for subculture.

**Binary Images:**

Double depictions have a completely unique because of this in MATLAB.A parallel photograph is a sensible cluster 0s and1s.Thus, a variety of 1s whose features are of measurements excellence, say unit8, and isn't always concept approximately as a twofold image in MATLAB. A numeric show off is modified to paired the utilization of spotlight coherent. In this manner, if A can be a numeric showcase along problem 1s, we make a cluster B using the announcement.

B=logical (A)

In the event that A contains of elements separated from 0s and 1s. Use of the intelligent capability changes over all nonzero segments to sensible 1s and all sections with rate 0 to coherent 0s. Utilizing social and valid administrators further makes clever well-known shows. To take a look at if a cluster is coherent, we make use of the I practical trademark: is logical(c). In the occasion that c is a coherent show off, this trademark restores a 1. Otherwise returns a zero. Consistent cluster is probably modified over to numeric reveals the utilization of the statistics style transformation presents.

**Indexed Images:**

Framework define a m\*3 kind of magnificence twofold containing skimming trouble esteems within the assortment [0, 1]. The duration m of the guide is identical to the huge sort of shades it characterizes. Each line of manual suggests the blood pink, green and blue brought materials of a solitary shading. Recorded pix make utilization of "coordinate mapping" of pixel electricity esteems shading map esteems. The tinge of every pixel is resolved through way of using the relating rate the whole range grid x as a pointer in to delineate. On the off danger that x is of modernity twofold, at that factor the majority of its segments with values masses substantially less than or indistinguishable to no less than one difficulty to the crucial column in delineate, brought materials with fee 2 thing to the second line et cetera. In the event that x is of complexity devices or unit 16, at that factor all delivered substances fee zero thing to the important line in outline, introduced materials with charge 1 aspect to the second et cetera.

**RGB Image:**

A RGB shading photograph is a M\*N\*three exhibit of tinge pixels wherein each coloration pixel is triplet much like the purple, inexperienced and blue brought materials of a RGB image, at a particular spatial area. A RGB image is probably considered as "stack" of three dim scale pics that after advocated in to the darkish pink, green and blue contributions of a tinge display screen. Deliver a shading picture at the show. Tradition the three previews shaping a RGB color image are alluded to as the red, unpractised and blue brought substances pictures. The information fashion of the brought materials images comes to a decision their form of qualities. On the off hazard that a RGB image is of modernity twofold the type of traits is [0, 1]. Correspondingly the sort of characteristics is [0,255] or [0, 65535]. For RGB pics of modernity gadgets or unit sixteen individually. The form of bits uses to speaks to the pixel estimations of the aspect pictures makes a decision the bit profundity of a RGB photo. For instance, if every aspect image is an 8bit picture, the evaluating RGB photo is expressed to be 24 bits profound. For the most part, the collection of bits in all inconvenience snap shots is the indistinguishable. For this case the type of feasible shading in a RGB photograph is (2^b) ^three, in which in b is numerous bits in the entirety about. For the 8bit case the amount is 16,777,216 colorations.

**Advantages of digital image:**

* The processing of images is faster and more cost-effective. One needs less time for processing, as well as less film and other photographing equipment. ƒ
* It is more ecological to process images. No processing or fixing chemicals are needed to take and process digital images. However, printing inks are essential when printing digital images.
* When shooting a digital image, one can immediately see if the image is good or not. ƒ copying a digital image is easy, and the quality of the image stays good unless it is compressed.
* For instance, saving an image as jpg format compresses the image. By resaving the image as jpg format, the compressed image will be recompressed, and the quality of the image will get worse with every saving.
* Fixing and retouching of images has become easier. In new Photoshop 7, it is possible to smooth face wrinkles with a new Healing Brush Tool in a couple of seconds.
* The expensive reproduction (compared with restoring the image with a repro camera) is faster and cheaper.
* By changing the image format and resolution, the image can be used in a number of media.

## Applications of Digital Image Processing:

Some of the major fields in which digital image processing is widely used are mentioned below.

* Image sharpening and restoration
* Medical field
* Remote sensing
* Transmission and encoding
* Machine/Robot vision
* Color processing
* Pattern recognition
* Video processing
* Microscopic Imaging