Journal of Interdisciplinary Mathematics
ISSN: 0972-0502 (Print), ISSN: 2169-012X (Online)
Vol. 27 (2024), No. 2, pp. 317–342
DOI : 10.47974/JIM-1830

Deep insight : Mathematical modeling and statistical analysis for mango leaf disease classification using advanced deep learning models

Priya Mathur [‡]
Department of Mathematics
Poornima Institute of Engineering & Technology
Jaipur
Rajasthan
India

Farhan Sheth [†]
Department of Computer Science & Engineering
Manipal University Jaipur
Jaipur
Rajasthan
India

Dinesh Goyal [§]
Department of Computer Science & Engineering
Poornima Institute of Engineering & Technology
Jaipur
Rajasthan
India

Amit Kumar Gupta *
Department of Computer Science & Engineering
Manipal University Jaipur
Jaipur
Rajasthan
India

[‡] *E-mail:* drpriyamathur21@gmail.com
[†] *E-mail:* farhansheth.jb@gmail.com
[§] *E-mail:* dinesh.goyal@poornima.org
* *E-mail:* dramitkumargupta1983@gmail.com (Corresponding Author)

Abstract

This paper presents a comprehensive investigation into the mathematical modeling and statistical analysis of classification of mango leaf diseases employing state-of-the-art Deep Learning models. The literature review underscores the significance of automated disease detection in agriculture, with Convolutional Neural Networks (CNNs) emerging as pivotal tools for image-based tasks. The proposed methodology encompasses meticulous preprocessing, optimal hyperparameter tuning, and fine-tuning of pretrained models (Inception V3, MobileNet V3 Small, MobileNet V3 Large, and ResNet50). Results indicate rapid convergence and outstanding accuracy during initial training, with all models achieving 100% accuracy on both validation and test datasets. K-Fold cross-validation affirms the models' consistency, with Inception V3 demonstrating leading performance. Detailed analyses, including training and loss graphs and confusion matrices, offer nuanced insights and highlight areas for refinement, particularly in distinguishing Healthy leaf, Gall Midge, and Anthracnose. This research positions the proposed methodology as a promising approach with potential applications in real-world agricultural scenarios, where precise disease detection is critical for effective crop management and optimal yields.

Subject Classification: *68T05, 68T07.*

Keywords: *Mathematical modeling, Statistical analysis, Mango leaf diseases Convolutional neural networks (CNNs), Image-based classification, Inception V3, MobileNet V3, ResNet50.*

1. Introduction

Agriculture stands at the forefront of global challenges, where ensuring food security and maximizing crop yield are imperative. In this context, the automated detection and classification of plant diseases play a pivotal role in precision agriculture, enabling timely intervention and effective crop management. Mango cultivation, a significant contributor to the agricultural sector, faces the threat of various diseases that can adversely impact yield and quality. This paper addresses the critical need for an accurate and efficient disease classification system for mango leaves. Leveraging recent advancements in deep learning, Convolutional Neural Networks (CNNs) have demonstrated remarkable capabilities in image-based tasks, particularly in the realm of disease detection. The proposed methodology integrates insights from the MangoLeafBD dataset [1] and draws upon the advancements presented in "LeafNet," a proficient CNN tailored for mango leaf disease detection [2]. The study encompasses a robust preprocessing pipeline involving color-to-grayscale conversion [3], Gaussian blur [4], and adaptive gamma correction [5], [6]. Subsequently, the dataset undergoes optimal enhancement to accentuate relevant features. Fine-tuning is applied to state-of-the-art CNN architectures,

including Inception V3, MobileNet V3 Small and Large, and ResNet50 [2]. The introduction of these advanced models aligns with the objective of achieving superior disease classification accuracy. This research not only contributes to the domain of agricultural technology but also addresses a pressing need for automated disease detection systems in mango cultivation, emphasizing the potential impact on global food security.

2. Related Study

The detection and classification of plant diseases have become increasingly critical for ensuring global food security. In recent years, the integration of advanced image processing techniques and deep learning models has shown promise in accurately identifying and diagnosing plant diseases. This literature review aims to provide an overview of the key research studies related to mango leaf disease detection, focusing on the MangoLeafBD dataset and the development of LeafNet, a proficient convolutional neural network (CNN). The study by Ali et al. (2022) introduced the MangoLeafBD dataset, a valuable resource for research in mango leaf disease detection. The dataset, available on Mendeley Data, provides a standardized collection of mango leaf images for training and testing machine learning models. This dataset forms the foundation for subsequent research endeavors, including the development and evaluation of LeafNet [1]. Rizvee et al. (2023) proposed LeafNet, a convolutional neural network specifically designed for detecting seven prominent mango leaf diseases. The model demonstrates high proficiency in accurately identifying diseases, showcasing the potential of deep learning in agriculture. This study contributes to the growing body of research on specialized CNN architectures for plant disease detection [2]. Several studies have explored image preprocessing techniques to enhance the quality of input images for disease detection models. Saravanan (2010) and Gedraite et al. (2011) investigated color-to-grayscale conversion and the effects of Gaussian blur in image filtering and segmentation, respectively. These techniques play a crucial role in optimizing image data for subsequent analysis [3-4]. Rahman et al. (2016) and Amiri & Hassanpour (2012) explored adaptive gamma correction as a preprocessing approach for image enhancement. These studies highlight the significance of gamma correction in improving image quality, which can positively impact the performance of disease detection models [5-6]. Xu et al. (2017) focused on Canny edge detection, while De Natale & Boato (2017) investigated morphological filtering of binary images. These techniques contribute to

feature extraction and image segmentation, essential steps in the development of accurate disease detection models [7-8]. Setiawan et al. (2013) and Reza (2004) explored the use of CLAHE for color retinal image enhancement. The adaptation of CLAHE for image enhancement is relevant to mango leaf disease detection, as it addresses issues related to varying image contrasts [9-10]. The literature review includes seminal works on deep learning, such as Krizhevsky et al. (2012), Deniz et al. (2018), and Hubel & Wiesel (1959), demonstrating the evolution of convolutional neural networks and their applications in various domains, including medical image analysis [11-13]. The study by Sharif Razavian et al. (2014) and others (Tajbakhsh et al., 2016; Gu et al., 2018) underscores the significance of transfer learning in image classification tasks, suggesting its potential application in mango leaf disease detection [14-16]. Optimization algorithms play a crucial role in training deep learning models. Kingma & Ba (2014) introduced the Adam optimization algorithm, while Li et al. (2020) reexamined hyperparameters for fine-tuning, contributing to the ongoing refinement of training strategies [17-18]. The inception architecture (Szegedy et al., 2016) and residual learning (He et al., 2016) represent architectural advancements in CNNs, emphasizing the importance of model design in achieving state-of-the-art performance in image recognition tasks [19-20, 24-27].

3. Methods and Material

The methodology adopted for this research is bifurcated into two key components. The initial phase outlines the preprocessing techniques applied to augment the dataset's suitability and interpretability for the model. Subsequently, the study delves into the deep learning models employed. Figure 1 illustrates the sequential steps involved in data

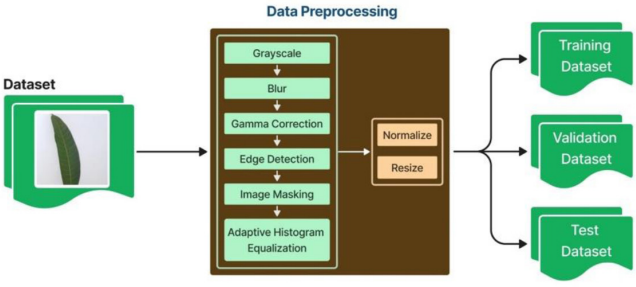

Figure 1

Sequential Steps involved in Data Preparation

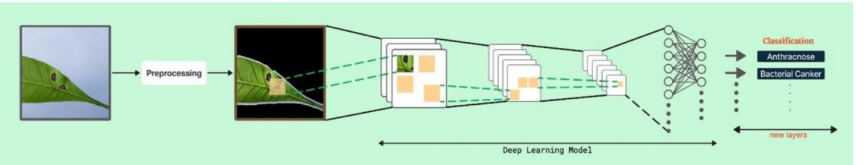

Figure 2

Deep Learning model workflow

preparation, as outlined in the roadmap. Additionally, Figure 2 offers a visual representation of the workflow associated with the deep learning model utilized in the study.

3.1 *Dataset Characteristics*

The mango leaf disease dataset employed in this study was obtained from MangoleafDB [1]. The images were sourced from mango trees in diverse regions across Bangladesh, encompassing eight classes. One class consists of normal mango leaf images, while the remaining seven classes represent various diseases: Anthracnose, Bacterial Canker, Cutting Weevil, Die Back, Gall Midge, Powdery Mildew, and Sooty Mould. Recognizing the potential bias and error introduced by background variations during training, the authors ensured uniformity by presenting images against a white background [1][2]. Additionally, the dataset underwent augmentation using fundamental techniques like rotation and zoom to enhance its diversity. With a well-balanced distribution, the dataset comprises a total of 4000 images, evenly distributed with 500 images per class. Figure 3 illustrates an example image from each class within the dataset.

Figure 3

Data Set Sample Images

3.2 *Data Preprocessing*

Preprocessing assumes a critical role in elevating data quality before model training. While the study's images featured a consistent white background, it's imperative to recognize that real-world scenarios may introduce challenges such as diverse backgrounds, noise, jitter, or fluctuations in brightness. Extracting pertinent information from images is paramount for enhancing reliability. The study's applied steps or algorithm facilitate the isolation of the leaf component in the image. Notably, images depicting diseased or pruned leaves may exhibit areas with disparate contrasts compared to the rest of the leaf. Leveraging a contrast enhancement method proves valuable in accentuating these specific regions. Effective data preprocessing is integral in readying datasets for model training, ensuring the model excels in discerning patterns and features amid varied environmental conditions. This contributes significantly to the robustness and generalizability of the trained model. Figure 4(a) visually represents the changes applied during preprocessing, while additional preprocessed samples from the dataset

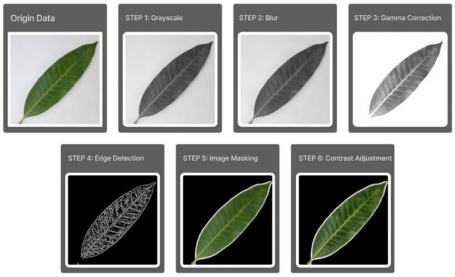

Figure 4(a)

Data Preprocessing Steps

Figure 4(b)

Preprocessed Samples from dataset

are showcased in Figure 4(b). The preprocessing steps further explained in below section.

3.2.1 Grayscale

Initially, the image undergoes conversion to a grayscale format, marking a foundational step that is essential for achieving optimal results in edge extraction and image segmentation. A grayscale image is defined by a single channel representing luminance, capturing intensity information across the entire spectrum of the image [3]. This initial transformation plays a crucial role in enhancing specific aspects of subsequent image processing tasks. The grayscale conversion process is articulated by the equation in (1), where R, G, and B denote the three-color channels—Red, Green, and Blue, respectively.

$$\text{RGB to Gray}: \; Y = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \tag{1}$$

3.2.2 Blur

To mitigate the visibility of numerous extraneous edges and ensure that only prominent edges are accentuated, blurring techniques come into play. In this study, we have employed Gaussian Blur, a technique akin to conventional average blurring, but with a weighted mean calculation rather than a simple average. In this context, neighborhood pixels closer to the central pixel contribute more weight, with the original pixel's value receiving the highest weight, while the surrounding pixels progressively receive lesser weight as the distance from the original pixel increases [4]. Gaussian smoothing serves the purpose of eliminating noise that approximately follows a Gaussian distribution, employing a kernel size of M × N, where both M and N are odd integers. The generation of the kernel involves utilizing a two-dimensional Gaussian function, as specified in equation (2) [4], where A is the amplitude, σ_x and σ_y are the standard deviations in the x and y directions, respectively, and the center is indicated by (p, q).

$$f(x,y) = A.e^{-\left(\frac{(x-p)^2}{2\sigma_x^2} + \frac{(x-q)^2}{2\sigma_y^2}\right)} \tag{2}$$

3.2.3 Gamma Correction

The image undergoes optimal enhancement via gamma correction, a widely adopted preprocessing method. This technique involves refining

the brightness levels in the image, or correcting the image's brightness, by employing a nonlinear transformation between the input values and the mapped output values [5]. Gamma correction, also referred to as the Power Law Transform, is executed through the equation specified in (3) [6]. This method is conveniently available in the scikit-image Python library, where the equation is applied after scaling each pixel to the range of 0 to 1.

$$o = I^{\gamma} \tag{3}$$

In the equation, "o" represents the output intensity, "I" denotes the input intensity, and γ is a non-negative parameter symbolizing gamma. This parameter governs the relationship between the input and output intensities. Varying the value of γ results in differently brightened images: for $\gamma < 1$, the histogram shifts towards the right, making the output image brighter than the input image; when $\gamma = 1$, there is no effect; and for $\gamma > 1$, the histogram shifts towards the left, rendering the output image darker than the input image.

3.2.4 Edge Detection

The dataset utilized in this study predominantly comprised images with a nearly clear white background. However, recognizing the divergence of real-world scenarios from such pristine conditions, where images may contain impurities, diverse backgrounds, noise, or jitter, it becomes crucial to enhance the model's reliability in practical applications. To achieve this, the focus was directed towards the pertinent component of the image—the leaf. Addressing this requirement, an edge detection method was employed to identify and isolate the edges of the leaf through masking. This effectively eliminated extraneous background elements and impurities. In this study, Canny edge detection was chosen and implemented using the OpenCV library in Python. Canny edge detection, a multi-scale algorithm, operates on the principle of selecting edge points through a threshold approach. The efficacy of the results is significantly dependent on the chosen threshold. In the Canny edge detection process, several steps are involved. Prior to edge detection, the image undergoes denoising via a 5 × 5 Gaussian filter described in equation (4) [7].

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4}$$

To obtain the first derivative (Gx) in the horizontal direction and the first derivative (Gy) in the vertical direction, the smoothed image is filtered with the Sobel kernel in both the horizontal and vertical directions. Subsequently, from these filtered images, the edge gradient and the direction of pixels are determined using equations (5) [7] and (6) [7]. Following this, non-edge pixels are eliminated through a process known as non-maximum suppression, resulting in the retention of only a few lines. The next step involves selecting the hysteresis threshold, requiring two thresholds as a range. Pixels with amplitudes surpassing the high threshold are designated as edge pixels, while those falling below are excluded. Notably, if a pixel's amplitude falls between the two thresholds, it is retained only if connected to a pixel exceeding the high threshold [7].

$$G = \sqrt{G_x^{\,2} + G_y^{\,2}} \tag{5}$$

$$\theta = \tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{6}$$

3.2.5 Image Masking

The detected edges encompass the leaf border and its major veins. Employing a masking technique involves filling the area inside these edges to create a mask. This mask, when applied to the original image, effectively highlights the leaf. The binary image of leaf edges can be filled using morphological filters, which primarily consist of two fundamental operators: erosion and dilation, along with a kernel known as the structuring element. This element is defined as a binary mask with a given shape and a reference point, and its shape determines the filter's effect on the image [8]. Morphological dilation enhances the visibility of objects and addresses small holes within them, resulting in thicker lines and larger filled shapes. On the other hand, morphological erosion eliminates extraneous pixels and slender lines, preserving only substantive objects. This process leads to thinner lines and smaller shapes, contributing to a refined representation. For a binary image, erosion and dilation are defined as follows [8]:

$$I \ominus S = \{x \, s.t. \; S_x \subseteq I\} \tag{7}$$

$$I \oplus S = \{x \, s.t. \; S_x^B \subseteq I \neq \varnothing\} \tag{8}$$

Here, "I" represents the image, "S" denotes the structuring element, "Sx" symbolizes the structuring element with the reference point at x, and "S_x^B" signifies the reflective rotation of S positioned in x. To fill or connect the edges in the binary image of leaf edges, the dilation process was employed on the binary image, utilizing an elliptical structuring element with dimensions 3 × 3. The resulting mask was then applied to the image using a Bitwise AND operation. This operation involves the addition of two different images, where the pixel to be displayed in the final image is selected using an AND operation on each corresponding pixel of the images.

3.2.6 Contrast Enhancement

Images depicting diseased or pruned leaves often exhibit regions with distinct contrasts compared to the rest of the leaf. To specifically accentuate these areas and other major parts of the leaf, contrast enhancement methods prove valuable as they may improve model performance. Histogram equalizations are commonly employed for contrast adjustment tasks. Pixel values in images can span a range, with brighter images favoring one end of the spectrum and darker ones the opposite. The histogram equalization method redistributes pixel values to make them more uniform. However, in cases where global contrast calculations are insufficient, leading to suboptimal results, this approach may not hold good value [9][10]. In response to this limitation, Adaptive Histogram Equalization (AHE) emerges as a refined approach. AHE divides the image into smaller blocks or "tiles," equalizing the histogram within each block. AHE leverages localized adjustments to enhance the edges within specific regions of the image. While AHE excels at local contrast improvement, it may inadvertently amplify noise within the image. To address this concern, a variant of AHE called Contrast Limited Adaptive Histogram Equalization (CLAHE) is utilized. CLAHE incorporates a contrast-limiting mechanism, introducing a "clip limit." Pixels exceeding this specified limit undergo clipping, redistributing them to other bins before the histogram equalization process. Subsequently, bilinear interpolation is applied to remove artifacts in tile borders [10]. The equations utilized in CLAHE are outlined below:

$$f_{i,j}(n) = \frac{(N-1)}{M} \cdot \sum_{k=0}^{n} h_{i,j}\ (k) \tag{9}$$

$$\beta = \frac{M}{N}\left(1 + \frac{\propto}{100}\ (s_{\max} - 1)\right) \tag{10}$$

Here, M and N represent the number of pixels and grayscales in each region, respectively. The term hi,j (n) for n (ranging from 0 to N - 1) provides the histogram of the (i, j) region, as defined in equation (9) [10], which gives the Cumulative Distribution Function (CDF) scaled by (N – 1) for grayscale mapping. The parameter α represents the clip factor in percentage, and smax is the maximum slope, as described in equation (10) [10], which gives the clip limit β.

3.2.7 Normalization

Normalization stands as another data augmentation technique applied in this study. In this process, a tensor image is normalized using the mean and standard deviation (std), ensuring that input features are consistently scaled and centered. This factor often leads to improved convergence during the training phase. Enhanced convergence not only reduces training time but also elevates the efficiency and effectiveness of the overall model training pipeline. Optimal values for mean and standard deviation, representing the 1st and 2nd order statistics per channel, result in the calculation of the z-score of the data channel by channel, commonly referred to as 'standardization'. Standardization, achieved through this normalization process, plays a pivotal role in enhancing the model's generalization capabilities across diverse types of images. The equation for this normalization process is:

$$output_{channel} = \frac{input_{channel} - mean_{channel}}{std_{channel}} \tag{11}$$

3.2.8 Data Splitting

The dataset underwent a meticulous partitioning into three distinct subsets: 75% for training, 15% for validation, and an additional 15% for testing purposes. This partitioning strategy is purposefully designed to ensure a robust evaluation of the model's performance across various stages of development. The training subset, constituting 75% of the dataset, forms the groundwork for the model to learn and adapt to the underlying patterns within the data. Simultaneously, the validation subset, comprising 15% of the dataset, offers an intermediate assessment during training to measure the model's generalization capabilities. The remaining 15% of the

dataset is set aside for the testing subset, playing a crucial role in evaluating the model's performance on unseen data and providing a reliable indicator of its real-world applicability. To mitigate the risk of overfitting, the study primarily implements K-fold validation. This approach involves dividing the dataset into K folds and iteratively using K-1 folds for training, reserving the remaining fold for validation. By systematically rotating the folds, each data point is used for validation exactly once. This not only guards against overfitting, where the model might perform exceptionally well on the training data but poorly on new, unseen data, but also provides a more robust estimation of the model's performance across different subsets of the dataset. The judicious use of K-fold validation enhances the study's credibility and ensures the model's effectiveness across diverse data scenarios. While the training and validation datasets were merged for K-fold validation, the test dataset remained untouched, ensuring a reliable and unbiased assessment of the model's performance on completely new and unseen data.

3.2.9 Image Resizing

To meet the specific input size criteria of Convolutional Neural Network (CNN) models, a preprocessing step was undertaken to resize the acquired images. Dealing with images of varying dimensions, as is often encountered in real-world scenarios, can be challenging for deep learning models. For the training dataset, a data augmentation technique called Random Resized Crop was implemented. This involved randomly selecting a portion of the image, followed by cropping and resizing operations to align with the specified input size requirements. This approach contributed to enhancing the diversity of the training data. In contrast, the validation and testing datasets underwent a simpler resizing process. Images in these subsets were resized without additional augmentation, preserving the original content while aligning with the required input dimensions of the CNN models. The resizing operation focused on two sets of dimensions: 256 * 256 and 299 * 299. These standardized sizes were chosen based on the specific requirements of the CNN models employed in this study. This preprocessing step ensured uniformity in input sizes across all datasets, facilitating effective training, validation, and testing of the CNN models.

3.3 Convolutional Neural Network

The ascent of Convolutional Neural Networks (CNNs) in image processing tasks has been remarkable, primarily attributed to their high

performance in various image-based applications [11]. CNNs, falling under the category of neural networks, derive their name from the incorporation of convolutional layers in their architecture, which excel at identifying local features within images. These models typically consist of four main layers—convolution, max pooling, fully connected, and output layer—stacked one over another. The adaptability of the CNN architecture is a key feature, allowing configurations tailored to specific task requirements. The CNN's versatility is evident as it can function both as a feature extractor and a classifier, with the fully connected layer performing classification based on high-level features obtained from preceding convolution and pooling operations [12]. In the convolutional layer, each node, tasked with detecting local structures in input channels, connects to a limited subset of spatially connected neurons. To identify similar local features, connection weights are shared among nodes, referred to as kernels. Consequently, a convolutional layer with 'n' kernels learns to detect 'n' local features, and their influence across input images results in 'n' feature maps. While this significantly increases computational complexity, it is mitigated by incorporating a pooling layer, which selects the maximum feature response within overlapping or non-overlapping local neighborhoods. This process discards the precise location of these maximum responses, effectively reducing the size of the feature maps. Notably, modern CNN architectures often replace the pooling layer with a convolution layer featuring a stride larger than 1 [13][14]. CNNs undergo training through back-propagation algorithms, utilizing a cost function defined by equation (12) [15]. This process involves iteratively adjusting the network's parameters to minimize the cost function, enhancing the model's ability to accurately classify and extract features from input images.

$$\mathcal{L} = \frac{1}{X}\sum_{i=1}^{X} \ell(\theta; q^{(i)}, o^{(i)}) \tag{12}$$

Where θ denotes all parameters of CNN, X are desired input-output relations $\{(p^{(n)}, q^{(n)});\ i \in [1, \ldots.. X]\}$ where $p^{(i)}$ is the i^{th} input data, $q^{(i)}$ its corresponding target label and $o^{(n)}$ the output of CNN [15]. The optimization process seeks to identify the best-fitting set of parameters, done so by minimizing the loss function, commonly achieved through the use of Stochastic Gradient Descent (SGD) to optimize the CNN network.

3.4 *Fine-Tuning of pretrained CNN Models*

Creating a custom Convolutional Neural Network (CNN) tailored to a limited-size dataset poses challenges, making the process potentially arduous and susceptible to issues like overfitting [14]. In response to these constraints, Transfer Learning emerges as a widely embraced solution for smaller datasets. This approach involves fine-tuning pretrained deep learning models, initially trained on expansive datasets, to suit the characteristics of the specific dataset [22]. In this study, four distinct pretrained models for transfer learning were employed: Inception V3, ResNet50, MobileNetV3 Small, and MobileNetV3 Large, all recognized for their efficacy in transfer learning tasks [23]. These models were initialized with their default ImageNet weights, and their last layer was replaced with a new custom fully connected layer, featuring neurons corresponding to the number of classes within the dataset, aligning with the classification requirements. The study focuses on the classification of mango leaf diseases, utilizing fine-tuning as the preferred method. In contrast to transfer learning, where only the new layer is trained while the original layers remain frozen, fine-tuning involves updating all layers, including the original ones, allowing for updates to all. Fine-tuning has demonstrated success across various applications, particularly when adapting models to custom datasets [14][16]. For the fine-tuning process, the Adaptive Moment Estimation (ADAM) optimizer was chosen. ADAM is an algorithm designed for first-order gradient-based optimization of stochastic objective functions, integrating adaptive estimates of lower-order moments. Combining the advantages of AdaGrad and RMSProp, ADAM proves effective in both online and non-stationary settings. Notably, its parameter updates remain invariant to gradient rescaling, exhibiting stepsizes bounded by the stepsize hyperparameter. Moreover, ADAM operates without the requirement of a stationary objective and demonstrates proficiency with sparse gradients [17]. The determination of optimal parameters, or hyperparameters, for the optimizer was a critical aspect addressed through an exhaustive exploration of the parameter space. The adjustment of learning parameters is indispensable for achieving optimal model outcomes and mitigating the risks of overfitting or underfitting, allowing for higher accuracy [18]. In this study, the “Grid Search” methodology was employed, systematically evaluating various combinations of hyperparameter values defined within a set. Although exhaustive, this method is effective as it thoroughly explores a spectrum of possibilities to pinpoint the configuration that optimizes model

performance, especially computationally feasible due to the smaller dataset size and availability of small models. The considered parameters encompassed crucial aspects such as learning rate, regularization strength, and weight decay. Additionally, the ADAM optimizer was applied across all models in the study, employing cross-entropy loss. The mathematical functions for both binary and multi-class cross-entropy loss are defined as follows:

$$L_{cross\ entropy} = -\sum_{i}^{N} y_i \log(p_i) \tag{13}$$

$$L_{cross\ entropy\ for\ multi\ class} = -\frac{1}{N}\sum_{i}^{N}\sum_{j}^{M} y_{ij} \log(p_{ij}) \tag{14}$$

In the given equations, y_i and y_{ij} represent the ground truth value, pi represents the predicted probability value for the i^{th} image from the classifier, p_{ij} is the probability that a sample i is of class j as assigned from the classifier, N represents the number of rows, and M is the number of classes. The optimizer works to update the network parameters such as weights to minimize the loss function. The study employs sophisticated models, and the fundamental structure of all employed models is depicted in Figure 5. Google's Inception V3, a prominent CNN architecture, is known for its high performance and accuracy, making it a preferred choice for transfer learning tasks. Inception V3 has 48 layers, operates on a 299×299-input size, and utilizes inception modules, enabling the network to capture features at multiple scales [13]. The RMSProp Optimizer is integrated into the network, and an optimization strategy involves reducing parameters by factorizing large convolution layers. The Inception part of the network employs a grid reduction technique, resulting in a faster and more accurate architecture, particularly effective for smaller datasets [19]. ResNet, developed by Microsoft Research, addresses the complexity of training deep neural networks by introducing a residual learning framework to tackle the challenge of vanishing gradients [20]. The ResNet50 architecture, with 50 layers and 16 blocks, stands as a testament to the efficacy of the residual approach.

MobileNets, developed by Google AI and Brain research teams, constitute a set of CNNs integrating platform-aware NAS and NetAdapt for network search and improvements [21]. In MobileNetV3, the fusion of MobileNetV2 linear bottleneck and inverted residual structure enhances layer efficiency, introducing lightweight attention modules and modified

swish nonlinearities [21]. The performance of the CNN models was evaluated on multiple metrics, including accuracy, precision, sensitivity, specificity, and F-score.

$$\text{Accuracy} = \frac{\text{number of correct prediction}}{\text{total number of samples}} \tag{15}$$

$$\text{Precision} = \frac{TP}{TN + FP} \tag{16}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{17}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{18}$$

$$F - \text{score} = \frac{2 \times \text{Precision} \times \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}} \tag{19}$$

The terms TP, TN, FP, and FN represent true positives, true negatives, false positives, and false negatives, respectively. Given the balanced class

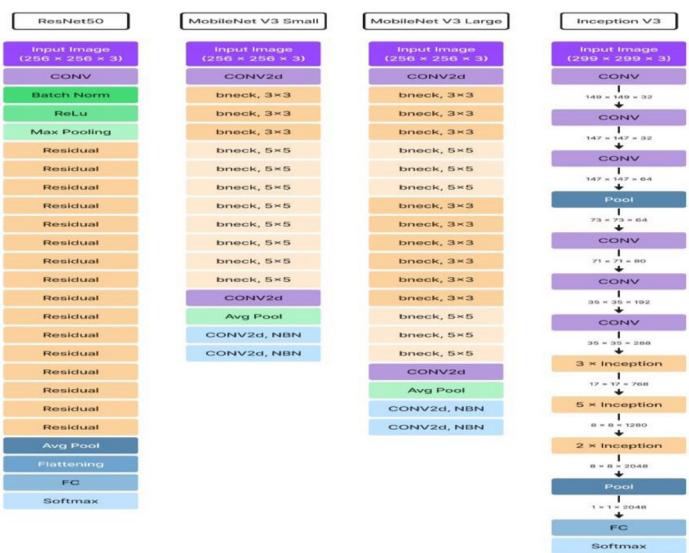

Figure 5

ResNet50, MobileNet V3 small, MobileNet V3 large, and Inception V3 architectures

distribution in the dataset, accuracy serves as a straightforward and effective metric for comparing the performance of different models. Additionally, the confusion matrix of all the models was employed as a metric, allowing an examination of common misclassifications among the models.

4. Result and Discussion

The Mango leaf disease dataset (MangoleafDB), comprising a total of 4000 images, was organized into 8 classes, each with 500 images, ensuring a balanced distribution. Following the methodology outlined above, the dataset was partitioned into three subsets: a training set with 75% of the images (2800 images), a validation set with 15% (600 images), and a test set with 15% (600 images). The training set maintained the balance, allocating 350 images per class, while the validation and test sets included 75 images per class.

4.1 *Optimal Hyperparameter*

For optimal training, hyperparameters were determined through a grid search approach. The lightweight model MobileNetV3 Small, configured as per Table 1, was selected for this process. The grid search focused on identifying the optimal learning rate and weight decay, as detailed in Table 2. The search involved evaluating all possible combinations of these parameters, resulting in nine sets. Each set, created from distinct combinations, took approximately 10 minutes to train, and the accuracy achieved for each combination is summarized in Table 3. After thorough evaluation, the set with the highest accuracy was identified, featuring a learning rate of 0.0001 and weight decay of 1e-08. This optimal configuration was chosen for training all the deep learning models. The final set of configurations, applicable to both individual model training and K-fold cross-validation, is presented in Table 4.

Table 1

Basic Configurations and Parameters for model for Grid Search

Configuration/Parameters	Value
Optimizer	ADAM
Batch Size	16
Max Epoch	30

Table 2

Parameters values used for Grid Search

Learning Rate	0.001	0.0001	1e-05
Weight Decay	1e-07	1e-08	1e-09

Table 3

Accuracy received on all sets of parameters in Grid Search

Learning Rate	Weight Decay	Accuracy (%)
0.001	1e-07	98.33
0.001	1e-08	99.0
0.001	1e-09	99.83
0.0001	1e-07	99.66
0.0001	1e-08	100
0.0001	1e-09	100
1e-05	1e-07	99.33
1e-05	1e-08	98.5
1e-05	1e-09	98.66

Table 4

Optimizer and Parameter settings

Sr. no	Parameter	Value
1	Optimizer	ADAM
2	Batch size	16
3	Maximum Epochs	30
4	Learning rate	0.0001
5	Weight decay	1e-08

4.2 *Fine-Tuning and Model Evaluation*

Following the preprocessing steps and employing optimal hyperparameters, Fine Tuning was applied to four distinct Deep Learning models: Inception V3, MobileNet V3 Small, MobileNet V3 Large, and ResNet50. The evaluation of these models involved a comprehensive comparison using various metrics, including accuracy, precision, sensitivity, specificity, and F-score. Given the balanced nature of the dataset, accuracy emerged as a reliable metric for comparison. Initially, the

models were trained with the specified parameters and dataset division, resulting in exceptional performance. As indicated in Table 5, all models swiftly converged, achieving 100% accuracy on both the validation and unseen test datasets. To further validate the models and optimize their performance, K-Fold cross-validation was conducted with a fold value of five (k = 5). The training and validation datasets were merged, while the test dataset remained unchanged, maintaining a proportion of 85% for training and 15% for testing. The overall performance of all models in 5-fold cross-validation is summarized in Table 5, with detailed fold-wise performance in Table 6. MobileNet V3 Small demonstrated the fastest training time, averaging 9 minutes for each fold, while Inception V3 required the longest, averaging 1 hour per fold. Despite varying training times, all models converged rapidly, achieving approximately 80% training accuracy and over 90% validation accuracy in the first epoch. Notably, MobileNet V3 exhibited an average accuracy of 70% in the first epoch. Training and loss graphs for all models are illustrated in Figure 6 and Figure 7. In terms of overall performance, Inception V3 outperformed other models, boasting the highest average K-Fold cross-validation accuracy of 100%. MobileNet V3 Large closely followed with an average K-Fold validation accuracy of 99.96%, while ResNet50 achieved 99.93%. MobileNet V3 Small exhibited the lowest average accuracy among all models, at 99.7%. The confusion matrices for models across all five folds are presented in Figure 8, 9, and 10 for MobileNet V3 Large, ResNet50, and MobileNet V3 Small, respectively. Misclassifications were predominantly observed among three classes: Healthy leaf, Gall Midge, and Anthracnose. The analysis confirms the effectiveness of the proposed methodology, delivering outstanding accuracy, with all models surpassing the 99.5% threshold.

Table 5

Mango leaf disease classification performance using 5-fold cross-validation.

Model	Accuracy (%)	Precision (%)	Sensitivity (%)	Specificity (%)	F-score (%)
ResNet50	99.93	99.935	99.93	99.990	99.93
MobileNet V3 Small	99.7	99.703	99.70	99.957	99.69
MobileNet V3 Large	99.96	99.967	99.96	99.995	99.96
Inception V3	100	100	100	100	100

Table 6

Mango leaf disease classification performance of each fold in 5-fold cross-validation

Model	Fold	Accuracy (%)	Precision (%)	Sensitivity (%)	Specificity (%)	F-score (%)
ResNet50	1	100	100	100	100	100
	2	100	100	100	100	100
	3	100	100	100	100	100
	4	99.66	99.67	99.66	99.95	99.66
	5	100	100	100	100	100
MobileNet V3 Small	1	99.5	99.50	99.5	99.92	99.49
	2	100	100	100	100	100
	3	99.83	99.83	99.83	99.97	99.83
	4	99.5	99.50	99.50	99.92	99.49
	5	99.66	99.67	99.66	99.95	99.66
MobileNet V3 Large	1	100	100	100	100	100
	2	100	100	100	100	100
	3	100	100	100	100	100
	4	99.83	99.83	99.83	99.97	99.83
	5	100	100	100	100	100
Inception V3	1,2,3,4,5	100	100	100	100	100

Figure 6

Accuracy and Loss graph for MobileNet V3 Small and ResNet50

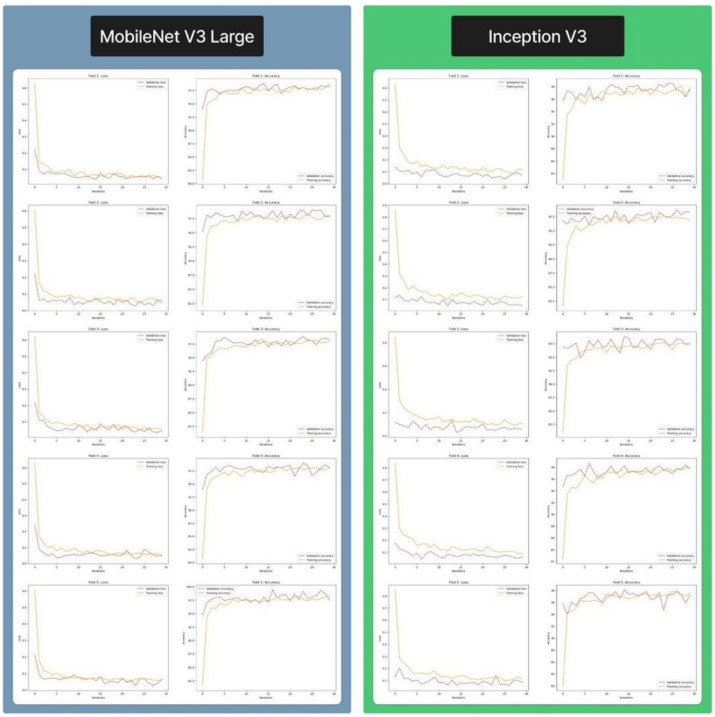

Figure 7

Accuracy and Loss graph for ResNet50 and Inception V3

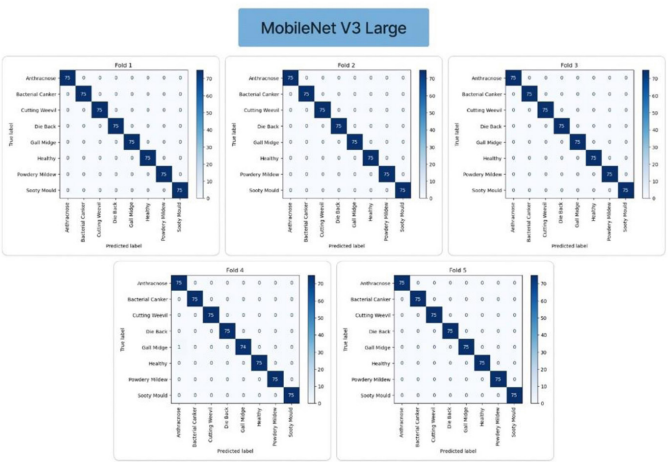

Figure 8

MobileNet V3 Large confusion matrix

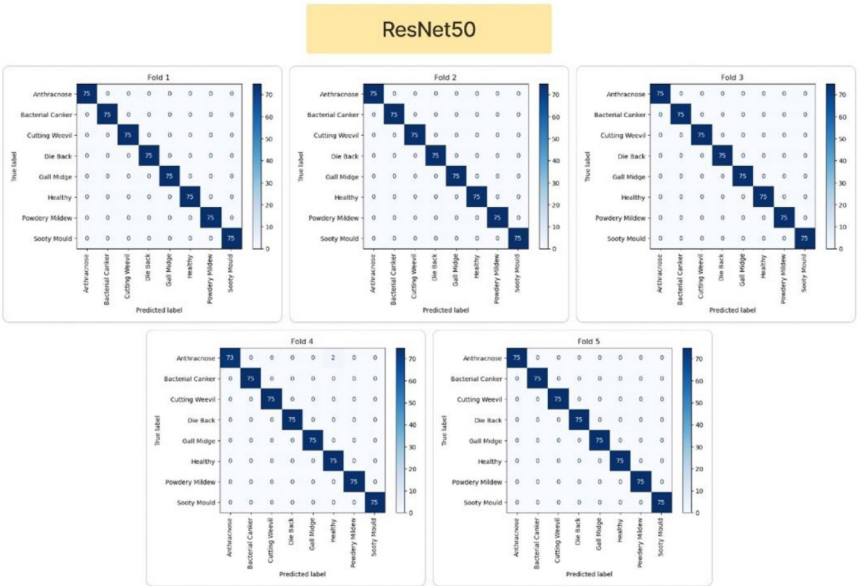

Figure 9

ResNet50 confusion matrix

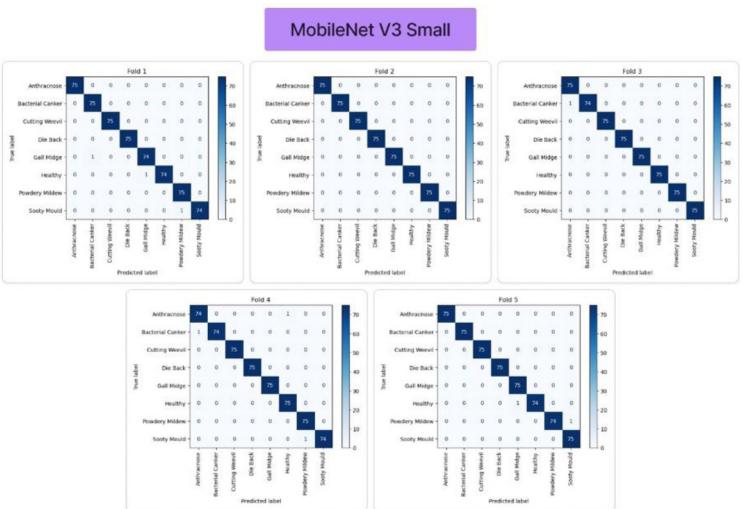

Figure 10

MobileNet V3 Small confusion matrix

The results of the study showcase the robust performance of the proposed methodology for mango leaf disease classification using Deep Learning models. After meticulous preprocessing and optimization with optimal hyperparameters, the fine-tuned models, including Inception V3, MobileNet V3 Small, MobileNet V3 Large, and ResNet50, exhibited remarkable accuracy. The initial training demonstrated swift convergence, achieving 100% accuracy on both validation and test datasets. Further validation through K-Fold cross-validation affirmed the models' consistency and efficacy, with Inception V3 leading the pack with a perfect average K-Fold accuracy of 100%. MobileNet V3 Large, ResNet50, and MobileNet V3 Small closely followed, achieving average accuracies above 99.5%. Training and loss graphs depicted rapid convergence, and despite variations in training times, the models in Figure 6-7 consistently demonstrated exceptional accuracy. Confusion matrices revealed that misclassifications were concentrated among specific classes, emphasizing shown in Figure 8-10 the need for targeted improvements in distinguishing Healthy leaf, Gall Midge, and Anthracnose. Overall, the study demonstrates the effectiveness and reliability of the proposed methodology in achieving high accuracy for mango leaf disease classification, laying a strong foundation for practical applications in agricultural diagnostics.

5. Conclusion

In conclusion, this paper presents a comprehensive exploration of mango leaf disease classification utilizing advanced Deep Learning models. The literature review provided insights into the significance of automated disease detection in agriculture and highlighted the role of Convolutional Neural Networks (CNNs) in image-based tasks. The proposed methodology, involving meticulous preprocessing, optimal hyperparameter tuning, and fine-tuning of pretrained models (Inception V3, MobileNet V3 Small, MobileNet V3 Large, and ResNet50), proved to be robust and effective. The results demonstrated exceptional accuracy during initial training, with all models converging rapidly and achieving 100% accuracy on both validation and test datasets. K-Fold cross-validation further validated the models' consistency, with Inception V3 leading in performance. The study's detailed analysis, including training and loss graphs and confusion matrices, revealed nuanced insights into model behavior and areas for potential refinement. Notably, misclassifications were concentrated in specific classes, prompting future work to enhance the models' ability to distinguish Healthy leaf, Gall Midge, and

Anthracnose. Overall, the proposed methodology showcases promising results, setting the stage for impactful applications in real-world agricultural settings, where accurate and efficient disease detection is crucial for crop management and yield optimization.

References

[1] Ali, Sawkat; Ibrahim, Muhammad ; Ahmed, Sarder Iftekhar ; Nadim, Md. ; Mizanur, Mizanur Rahman; Shejunti, Maria Mehjabin ; Jabid, Taskeed (2022), "MangoLeafBD Dataset", Mendeley Data, V1, doi: 10.17632/hxsnvwty3r.1

[2] Rizvee, R. A., Orpa, T. H., Ahnaf, A., Kabir, M. A., Rashid, M. R. A., Islam, M. M., ... & Ali, M. S., LeafNet: A proficient convolutional neural network for detecting seven prominent mango leaf diseases. *Journal of Agriculture and Food Research*, 14, 100787 (2023).

[3] Saravanan, C., Color image to grayscale image conversion. In 2010 Second International Conference on Computer Engineering and Applications, Vol. 2, pp. 196-199 (2010, March). IEEE.

[4] Gedraite, E. S., & Hadad, M., Investigation on the effect of a Gaussian Blur in image filtering and segmentation. In Proceedings ELMAR-2011, pp. 393-396 (2011, September). IEEE.

[5] Rahman, S., Rahman, M. M., Abdullah-Al-Wadud, M., Al-Quaderi, G. D., & Shoyaib, M., An adaptive gamma correction for image enhancement. *EURASIP Journal on Image and Video Processing*, 2016(1), 1-13 (2016).

[6] Amiri, S. A., & Hassanpour, H., A preprocessing approach for image analysis using gamma correction. *International Journal of Computer Applications*, 38(12), 38-46 (2012).

[7] Xu, Z., Baojie, X., & Guoxin, W., Canny edge detection based on Open CV. In 2017 13th IEEE international conference on electronic measurement & instruments (ICEMI), pp. 53-56 (2017, October). IEEE.

[8] De Natale, F. G., & Boato, G., Detecting morphological filtering of binary images. *IEEE Transactions on Information Forensics and Security*, 12(5), 1207-1217 (2017).

[9] Setiawan, A. W., Mengko, T. R., Santoso, O. S., & Suksmono, A. B., Color retinal image enhancement using CLAHE. In International conference on ICT for smart society, pp. 1-3 (2013, June). IEEE.

[10] Reza, A. M., Realization of the contrast limited adaptive histogram equalization (CLAHE) for real-time image enhancement. *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology*, 38, 35-44 (2004).

[11] Krizhevsky, A., Sutskever, I., & Hinton, G. E., Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25 (2012).

[12] Deniz, E., Şengür, A., Kadiroğlu, Z., Guo, Y., Bajaj, V., & Budak, Ü., Transfer learning based histopathologic image classification for breast cancer detection. *Health information science and systems*, 6, 1-7 (2018).

[13] Hubel, D. H., & Wiesel, T. N., Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3), 574 (1959).

[14] Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., & Liang, J., Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, 35(5), 1299-1312 (2016).

[15] Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., ... & Chen, T., Recent advances in convolutional neural networks. *Pattern recognition*, 77, 354-377 (2018).

[16] Sharif Razavian, A., Azizpour, H., Sullivan, J., & Carlsson, S., CNN features off-the-shelf: an astounding baseline for recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp. 806-813 (2014).

[17] Kingma, D. P., & Ba, J., Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014).

[18] Li, H., Chaudhari, P., Yang, H., Lam, M., Ravichandran, A., Bhotika, R., & Soatto, S., Rethinking the hyperparameters for fine-tuning. arXiv preprint arXiv:2002.11770 (2020).

[19] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z., Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818- 2826 (2016).

[20] He, K., Zhang, X., Ren, S., & Sun, J., Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778 (2016).

[21] Howard, A., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., ... & Adam, H., Searching for mobilenetv3. In Proceedings of the IEEE/CVF international conference on computer vision, pp. 1314-1324 (2019).

[22] Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., & Liu, C., A survey on deep transfer learning. In Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27, pp. 270-279 (2018). Springer International Publishing.

[23] Kornblith, S., Shlens, J., & Le, Q. V., Do better imagenet models transfer better?. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 2661-2671 (2019).

[24] Amit Kumar Gupta, Pushpa Gothwal, Dinesh Goyal & Carlos M. Travieso-Gonzalez. IoT-Galvanized pandemic special E-Toilet for generation of sanitized environment, *Journal of Discrete Mathematical Sciences and Cryptography* (2022), DOI: 10.1080/09720529.2022.2068607.

[25] Joshi, Ruchi, Mathur, Priya, Gupta, Amit Kumar, Singh, Suyesha, Paliwal, Vismita & Nayar, Sejal. Mathematical modeling of intelligent system for predicting effectiveness of premenstrual syndrome, *Journal of Interdisciplinary Mathematics*, 26:3, 551-562 (2023), DOI: https://doi.org/10.47974/JIM-1681.

[26] Kumar, Anil, Gupta, Amit Kumar, Panwar, Deepak, Chaurasia, Sandeep & Goyal, Dinesh. Operating system security with discrete mathematical structure for secure round robin scheduling method with intelligent time quantum, *Journal of Discrete Mathematical Sciences and Cryptography*, 26:5, 1519–1533 (2023), DOI: https://doi.org/10.47974/JDMSC-1816.

[27] Amit Kumar Gupta, Vijander Singh, Priya Mathur & Carlos M. Travieso-Gonzalez. Prediction of COVID-19 pandemic measuring criteria using support vector machine, prophet and linear regression models in Indian scenario, *Journal of Interdisciplinary Mathematics* (2020), DOI: https://doi.org/10.1080/09720502.2020.1833458.