IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 42, No. 6, December, 2025, pp. 3501-3528
Journal homepage: http://iieta.org/journals/ts

A Deep Learning Analysis on Training Issues of Turmeric Leaf Early Disease Detection

Ranjith Kumar Banala, Rajesh Duvvuru*

School of Computer Science and Engineering, VIT-AP University, Guntur 522241, India

Corresponding Author Email: rajesh.duvvuru@vitap.ac.in

https://doi.org/10.18280/ts.420636

Received: 11 March 2025
Revised: 21 August 2025
Accepted: 26 September 2025
Available online: 31 December 2025

Keywords:
deep learning, GoogLeNet, SqueezeNet, ResNet-50, Curcuma longa

ABSTRACT

The demand for medicinal herbs, especially Curcuma longa (Turmeric), has surged because to the COVID-19 pandemic, with study increasingly depending on AYUSH-based therapies. India, a prominent producer of Curcuma longa, is witnessing a reduction in crop yield due to climate factors. This requires early disease diagnosis for both veterinary and therapeutic applications. Deep learning, an advanced image-based object detection method, attains 100% accuracy on three disease datasets: leaf blotch, Colletotrichum leaf spot, and Cercospora leaf spot images. Three varieties of Deep Learning networks, SqueezeNet, GoogLeNet, and ResNet-50, are employed for parameter training, optimization techniques, and vector distance methodologies. The experiments demonstrated that all three Deep Learning techniques attained 100% training accuracy over many instances. The total performance of ResNet-50 with SDGM and ADAM surpassed that of SqueezeNet and GoogLeNet. GoogLeNet is recommended to surpass SqueezeNet in the majority of instances. The research underscores the necessity for enhanced training and validation techniques to attain elevated training accuracy in Curcuma longa.

1. INTRODUCTION

In the current global climate change scenario, we prefer technology-based solutions for the highest crop yielding rate. Current advanced technocratic approaches such as Artificial Intelligence and IoT are two popular areas that help traditional agriculture practices for food security and health informatics. Especially agriculture-based countries like India, China, the USA, and Brazil need efficient, smart mechanisms for sustainable food and medicinal production. Promoting health and well-being is identified as one of the seventeen global goals of the 2030 Agenda for Sustainable Development [1], as stated in Sustainable Development Goal (SDG) 3 of the United Nations. SDG #3. which pertains to health aims to establish universal health and well-being [2]. This includes a resolute dedication to eliminating epidemics of infectious diseases such as AIDS, tuberculosis, and malaria by the year 2030. Furthermore, it strives to establish Universal Health Coverage (UCH) and guarantee universal access to safe and efficacious vaccines and medications [3].

It is essential to support vaccine research and development and provide affordable medication access for this process to function. To achieve SDG3, currently, there is a need to introduce advanced technological practices for higher medicinal crop production. So far, many researchers and technocrats invented smart technology for easy agriculture practices such as precision agriculture, Crop Health Analysis, Climate-smart advisories, Yield forecast, Crop Damage Assessment, Supply Chain Management, Seed Production, Agro-Big Data Analytics, and smart farming practices. In addition to natural disasters and irregular weather patterns, medicinal plant diseases are another major cause of insufficient medicine production in the world [4]. The World Health Organization reports that 80% of the global population uses botanical remedies to fulfill at least a portion of their fundamental healthcare needs. Approximately 21,000 plant species have the potential to be utilized in medicinal applications, per the WHO [5]. It is a known fact that India and China largely depend on herbal treatment. Recently the Indian Govt. has proposed the concept of 'Integrated Medicine', where the Allopathic and AYUSH departments will work together to provide the best treatment that can be offered in one place [6]. At present the world is moving towards the usage of herbal medicines due to the less side-effect. Moreover, herbal medicines have become very popular in recent times due to wide usage in the treatment of COVID-19 (SARS-CoV-2). Especially the usage of Phyllanthus emblica, solanum indicum, Solanum surattense, Terminalia bellirica, Ficus religiosa, Piper longum, Curcuma longa Alhagi camelorum etc., are a few herbal medicinal plant species that are very frequent in COVID-19 treatment [7]. Among all other medicinal plants, Curcuma longa (Turmeric) is one of the herbal medicines that is used extensively in antibiotic preparation [8]. Cancer, rheumatoid arthritis, dermatitis, skin cancer, wound healing, urinary tract infections, and liver diseases are just some of the things that it can help with, Curcuma longa (Turmeric) is also employed in the treatment of these conditions [9, 10]. India is the leading producer and exporter of Curcuma longa (Turmeric) globally and shares 86% of global production. Among the Indian states, the state of Telangana stands first place for the Curcuma longa (Turmeric) [11, 12]. The state of Andhra Pradesh is one of the

foremost contributors to the overall production of Curcuma longa (Turmeric) in the country after Telangana state. Among the 24 finest varieties in the countries, the Duggirala variety has a special purpose with rich medicinal values. Figure 1 shows the production and average price of the Duggirala variety in the state of Andhra Pradesh. The statistical results show the demand for Curcuma longa (Turmeric) is very high irrespective of the production. The production of Curcuma longa (Turmeric) is reduced due to COVID-19 on the global export and import of goods and unfavorable climatic conditions. This study focuses on the early detection of diseases in Curcuma longa leaves. This study aimed to determine the most effective Deep Learning (DL) training network that can achieve 100% accuracy when applied to the disease dataset of Curcuma longa (Turmeric- Duggirala variant). The collection consists of photographs depicting leaf blotch, Colletotrichum leaf spot, and Cercospora leaf spot. The deep learning mechanism is one of the most advanced technologies that can accurately identify items based on how they appear in photographs. Two stages are included in the deep learning approaches, which are training and validation. It is generally accepted that the impact of the correctness of the data training would be reflected in the validation metric, which is known as the mean correctness Precision (mAP). For the most part, when the accuracy of the training is lower, the validation accuracy scores are significantly lower. Therefore, achieving a training accuracy of one hundred percent is a challenging challenge for increased mAP. Typically, the performance of the training method is categorized into Gain and loss functions. When the Gain achieves 100% accuracy, it enables the classifier to identify objects with greater confidence levels. When working with smaller datasets that are balanced, it can be especially challenging to determine the appropriate values for training parameters, such as the learning rate and the number of epochs. This study also examined the most effective training optimizers for training the smaller diseased Duggirala Curcuma longa (Turmeric) leaf dataset, including Stochastic Gradient Descent (SGDM), Root Mean Squared Propagation (RMSProp), and Adaptive Moment Estimation (ADAM), along with vector distance (L2Norm, Global-L2Norm, and Absolute) to achieve higher levels of accuracy.

Figure 1. Duggirala turmeric variant production and average price during 2017-2023
Data Source: AP-Turmeric yard, Duggirala

The present study investigated the optimal Deep Learning (DL) training network for attaining 100% accuracy on the illness dataset of Curcuma longa (Turmeric-Duggirala variation). The collection comprises photos of leaf blotch, Colletotrichum leaf spot, and Cercospora leaf spot. Training parameters such as Learning Rate, Epochs, Gradient Threshold Method, etc., play a crucial role in object detection for deep learning architectures. In most of the cases, the dataset is segregated into 70% for training and 30% for testing. During classification, the performance of the algorithm is based on trained data by specifying a Gain function. The nature of datasets is always dynamic and it is very difficult for the researcher to identify the right parameters to train and classify the datasets. Moreover, choosing the right parametric values for higher classification training and classification is always a challenging task. The discussion in more depth can be found in the sections that follow.

Duggirala Curcuma longa (Turmeric) Dataset: A novel, custom-balanced Duggirala Curcuma longa (Turmeric) dataset collected 885 raw images that contain three classes of diseased leaves, such as leaf blotch, Colletotrichum leaf spot, and Cercospora leaf spot. Later dataset size was increased to 6,584 using pre-processing annotation techniques like image flip, rotation etc. and generated with balanced classes. The Duggirala Curcuma longa (Turmeric) Dataset is distinct and has not been previously available.

• Higher Training Accuracy: The results produced a higher training accuracy on smaller and balanced Duggirala Curcuma longa (Turmeric) Dataset with limited classes by tuning hyper parameters such as Learning rate, Epochs etc. This work achieved improved accuracy by systematically adjusting the hyperparameters of the optimizer and architecture.

• Impact of Optimizer on Training Accuracy: This work presents fresh findings on the influence of an optimizer on different widely-used deep learning approaches when applied to smaller, balanced datasets. This task is particularly tough, especially when dealing with damaged leaf data. This study specifically examines the performance of optimizers, including SGDM, RMSProp, and ADAM, in training neural networks. Our findings indicate that utilizing the RMSProp optimizer yields the highest classification accuracy among all designs while using lower learning rates and higher epochs.

• Impact of Vector Distance Methods on Classification Accuracy: This study presents a unique technical analysis by combining deep learning architectures with hyperparameters and optimizers on the Duggirala Curcuma longa (Turmeric) Dataset concerning the vector distance methods (L2Norm, Global-L2Norm, and Absolute) The study demonstrates the effectiveness of this approach by achieving higher accuracies on various common deep learning architectures and presenting multi-dimensional results. Based on vector distance methods the technical examination, defines that the ResNet-50 is the most suitable deep learning architecture for smaller datasets such as Duggirala Curcuma longa (Turmeric).

This work mostly concentrated on training-related concerns exclusively. Training the dataset is crucial for attaining superior test accuracy. The majority of research emphasizes end classification accuracy; however, if lesser test accuracy is achieved, a primary reason may be insufficient training accuracy resulting from suboptimal optimizer selection and hyperparameter configuration. Training accuracy is the paramount idea essential for achieving superior categorization in deep learning architectures. However, most research did not tackle the problem of inadequate training concerns. This paper examines training challenges with widely-used optimizers, including SGDM, ADAM, and RMSProp.

The current study focuses on conducting a comprehensive

comparative analysis of deep learning architecture, optimizers, and hyper-parameters to determine the most effective training classification method, particularly for smaller datasets such as Duggirala Curcuma longa (Turmeric), which is distinct. These types of research are scarce in the data science literature, which typically examines a broad variety of characteristics and provides concise discussions. The current work just focuses on training accuracy and does not address validation accuracy in relation to the suggested dataset. Although the training accuracy hits 100%, there is uncertainty on whether the test accuracy can achieve 99%. The ultimate goal is to detect diseases in the leaf dataset; hence validation is a crucial aspect of the performance evaluation.

The subsequent sections of this study are structured as follows. Background research in the fields of agriculture and deep learning architectures is detailed in Section 2, early disease detection and training issues, and a novel image dataset of the Duggirala Curcuma longa (Turmeric) variant is prepared and discussed in Section 3. Similarly, Section 4 describes the mathematical notations and assumptions regarding explains details regarding methodology with the mathematical representation of gradient methods like RMSProp, ADAM, and L2 Regularization. Then section 5 describes the experimentation environment setup and section 6 explains the deep learning architecture. Then section 7 discusses the experiments of the models, Section 8 represents the results. Similarly, Section 9 detailed the performance evaluation metrics, then Section 10 represents the Anova test for the optimizers and then discussion. Lastly, the paper is summarized and suggests for best training classification methods in conclusions.

2. RELATED WORKS

Early disease detection is one of the important research areas, where researchers are trying to achieve 100% classification accuracy in disease detection. Orchi et al. [13] presented a detailed study on the AI and IoT approaches for crop disease detection and the survey revealed 97% as the highest classification accuracy. Thangaraj et al. [14] worked the disease detection on the Tomato leaf disease dataset and their investigation showed that 52% of present leaf disease detection techniques classification accuracy is 90% and 48% of present studies area less than 90%. The study also revealed that AlexNet disease detection resulted in 98.6% classification accuracy and also investigated on GoogLeNet (99.18%), ResNet-50 (98.8%), Xception (98.13%), and VGG16 (99.25%) as maximum classification accuracy. However, it is observed that none of the deep learning architecture achieves a cent percent accuracy. Math and Dharwadkar [15] used a deep convolution network for early disease detection on grape disease datasets and the classification model resulted in 99.34%. Naeem et al. [16] identified medicinal plant leaves based on multispectral and textural characteristics using machine learning; they obtained the following percentages: 98.40% for Catnip, 99.80% for Peppermint, 99.10% for Tulsi, 98.40% for Bael and 99.20% for Stevia, 99.90% for Lemon balm. Arunaggiri Pandian et al. [17] have proposed disease detection in medicinal plants using Convolutional Neural Network (CNN) approaches such as ResNet101, InceptionV3, and VGG16 and resulted in 97.32% classification accuracy by InceptionV3 on the Ayur Bharat dataset. In collaboration with the Central Council for Research in Ayurvedic Sciences (CCRAS), and Indian Council of Medical Research (ICMR) the Ministry of AYUSH, and GoI, this CTRI Ayurveda Dataset was compiled. Based on the Deep Herb dataset, Roopashree and Anitha suggested an autonomous medicinal plant identification system employing several neural network approaches in computer vision and deep learning. The Deep Herb contains 2515 leaf photos from 40 different Indian herb species. The Deep Herb model learned by Xception and ANN outperformed the research by 97.5% [18]. Kuricheti and Supriya [19] examined the necessity and application of intelligent agricultural technologies by detecting and classifying turmeric leaf diseases via computer vision. Gogoi et al. [20] constructed a novel Curcuma longa (Turmeric) dataset for leaf detection. Chen et al. [21] applied a unique AgriTalk technique on a Turmeric smaller dataset that predicts the quantity of Bacillus based on modern IoT and machine learning technologies with mean absolute percentage errors (MAPEs) ranging from 6.73% to 19.76%. Devisurya et al. [22] used deep learning approaches like YOLOv3 methods and Faster R-CNN with the VGG16 model on the turmeric dataset for early disease detection and achieved 83.4% classification accuracy. To obtain 100% accuracy the training parameters must contain the Learning rate, Epochs, Gradient Threshold method, batch learning method, and gain and loss of dataset play a crucial role, where training parameters like the gradient techniques, such as, ADAM, Root Mean Square Propagation (RMSProp) and Stochastic Gradient Descent with Momentum (SGDM) are widely used algorithms for mini-batch processing.

3. DUGGIRALA CURCUMALONGA DATA SET

This study collects unique images of medicinal plant leaves of Curcuma longa of Duggirala variant. The Curcuma longa medicinal plant leaf dataset contains the Curcuma longa leaf blotch, the Colletotrichum leaf spot, and the Cercospora leaf spot. Three types of diseased leaves are gathered from the field, and the images of Curcuma longa are divided into three different classes. The Ayur Bharat and Deepherb is popular Indian medicinal plants databases which is open soruce This database establishes a comprehensive database encompassing 6,959 medicinal plants distributed across 28 states and 8 union territories, detailing their phytochemical properties and geographical distribution. It derives information from traditional knowledge, geographical indications, phytochemicals, and chemoinformatics.

The Ayur Bharat and Deepherb databases have limited turmeric images, and the Duggirala variation is unavailable. This study experiments on the innovative leave dataset obtained during fieldwork at the Crop area and Turmeric Board in Duggirala, Guntur District, Andhra Pradesh, India. HD 1024×768 pixels are used in the experiment. Clean the picture dataset for better categorization. Segmentation, picture scaling, contrast improvement, color correction, noise reduction, and feature extraction were used to preprocess the dataset.

To improve the classification analysis, the current Curcuma longa leaf dataset is analyzed using several color band indices that enhance ambient light conditions for color correction. Color band indices are commonly employed in crop image processing studies and are calculated by assessing the ratio of light intensity across different radiation bands. Light intensity is often averaged throughout the entire image or a designated segment of it.

The computation of the ratio of color band intensities

functions as a normalization step to adjust for variations in ambient light conditions. This current Curcuma longa leaf picture dataset is analyzed using multiple Excess Red Index and Excess Green Minus Index (ERIEGMI) [23], Excess Blue color index (EBI), Excess Red Index (ERI) and Excess Green Index (EGI). ERI, EBI and EGI are improved the Classification analysis for Curcuma longa leaf images dataset samples.

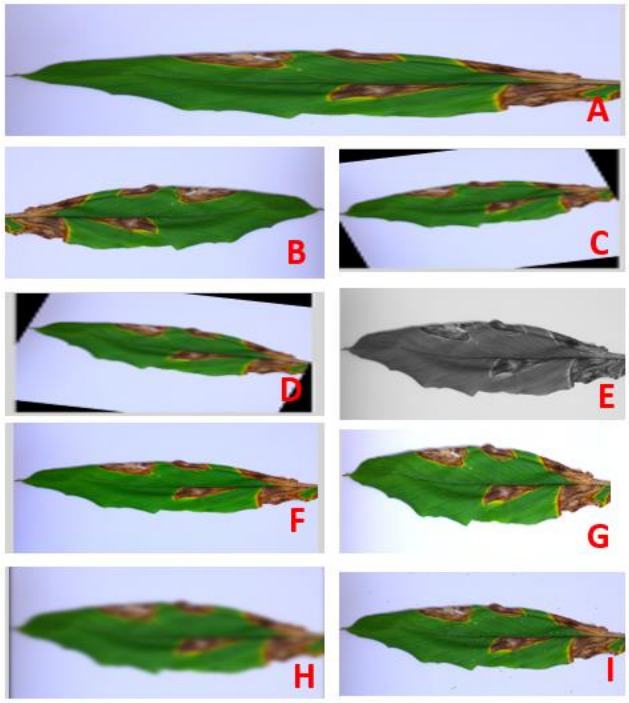

Figure 2. (A) Original image; (B) Flip (Horizantal); (C) Rotate (-15˚); (D) Rotate (15˚); (E) Grayscale (15%); (F) Saturation (25%); (G) Brightness (15%); (H) Blur (Upto 2.5px); (I) Noise (Upto 0.1% of pixels)

Mainly three categories of sick leaves are included. Curcuma longa plant, popularly known as the Turmeric plant, is particularly subject to three diseases that greatly harm its rhizomes. The collection includes 6584 annotated photographs of leaf blotch, Colletotrichum leaf spot, and Cercospora leaf spot. The data is obtained using a high-resolution camera (Canon EOS 5DS R with 50.6 megapixels) from a specific area of land in Duggirala Mandal, Guntur District, Andhra Pradesh, India. Subsequently, the data undergoes picture pre-processing processes to create a dataset of superior quality. Pre-processing eliminates unwanted distortions and boosts important features necessary for the intended purpose. The noise data eliminated from the raw photos, categorized by class. In the production of the dataset, we utilize the following pre-processing techniques: (i) *Data Profiling*: The Curcuma longa dataset underwent profiling stages, which involved analyzing the distribution of colors, sizes, brightness, and shuffling. The open-source tool 'Data Gradients' is used to examine the Curcuma longa dataset. The analysis focuses on many criteria related to picture quality, such as convexity, fine features, segments, brightness, color distribution, aspect ratios, and resolution of leaf photographs. Figure 2 illustrates the Augmented techniques together with their parameters, while the distribution of three classes within the dataset, indicating that the dataset consists of 885 raw photographs classified into three categories, which were then increased to 6,585 images after annotation. (ii) *Data Cleaning*: We eliminated erroneous, corrupted, improperly formatted, redundant, or unfinished photos of Curcuma longa from the dataset. A total of 292 images of Leaf blotch were acquired, with 18 photographs being excluded from the collection. Similarly, I captured 305 shots of Colletotrichum leaf spot and then removed 29 of them. The majority of the discarded images were duplicates, while just a small number were found to be blurry. Following the same pattern as the previous instance, I gather a total of 288 photographs of Cercospora leaf spot. However, I exclude 15 of them as they possess inadequate resolution and brightness. We remove 62 anomalous data points from the original sample of 885, resulting in a balanced dataset of 885 photographs [24]. (iii) *Balanced Dataset*: Classifying balanced datasets yields much higher accuracy and reduces bias compared to imbalanced datasets. For the current research, a selection of 885 high-quality photos of pests was meticulously picked and resized to a resolution of 640 by 640 pixels in each category. Subsequently, these images undergo amplification [25]. (iv) *Augmentation*: Each pest class image undergoes a five-fold augmentation process, which involves rotation (four times), blurring (once), adding salt-pepper noise, and flipping (twice). This results in a total of 6584 augmented images across all three classes. The dataset is partitioned into three groups, allocating 20% for validation, 10% for testing, and 70% for training. The dataset distribution chart displays the distribution of Curcuma longa (Turmeric) leaf disease classes, which include three variants: Duggirala variation Leaf blotch (2192), Colletotrichum leaf spot (2208), and Cercospora leaf spot (2184).

Dataset

A total of 885 raw images of all classes were collected from the field. Among collected Colletotrichum leaf spot image took major portion (305) and followed by Leaf blotch (292), Cercospora leaf spot (288). A Data Training experimentation also conducted on imbalanced dataset and obtain 38 percent training accuracy for Squeeznet,29 percent for GoogLeNet and 42 percent for ResNet-50, when Hyper-parameters such as Learning rate (0.01), Epochs (25), Optimizers is SGDM and vector distance method is L2norm performs low accuracy with imbalance dataset. So, the rest of the work is discussed only on balanced datasets rather than imbalanced data set that resulted poor accuracy for all three training networks

4. METHODOLOGY

The methodology comprises the popular deep learning architectures SqueezeNet, GoogLeNet, and ResNet-50 to train turmeric leaf data, which helps to attain higher testing accuracy in early warning of diseases. With the addition of the turmeric image dataset, the dataset has undergone a data cleaning process. The dataset is separately subjected to a deep learning architecture. Compute the training classification accuracy by adjusting the training parameters, such as learning rate, epochs, gradient threshold methods, and other parameters as constants. Table 1 shows the Deep Learning Layer Architecture Comparative Analysis of SqueezeNet, GoogLeNet, and ResNet-50.

By adjusting the parameters and altering the training methods, we calculate the accuracy gain or loss values, ultimately determining the training accuracy. Similarly, we

calculate and suggest all three architecture training accuracies, aiming for the best accuracy and specified training parameter values. Layered structure is shown in Figure 3.

Table 1. Deep learning layer architecture comparative analysis

S. No.	Layer Name	SqueezeNet		GoogLeNet		ResNet-50	
		No.	Y/N	No.	Y/N	No.	Y/N
1	Input	1	√	1	√	1	√
2	SoftMax	1	√	1	√	1	√
3	Global Average Pooling 2D	1	√	1	√	1	√
4	Dropout	1	√	1	√	0	X
5	ReLU	26	√	57	√	48	√
6	Fully Connected	1	√	1	√	1	√
7	Max Pooling 2D	3	√	14	√	1	√
8	Convolution 2D	26	√	57	√	53	√
9	Depth Concatenation	8	√	9	√	0	X
10	Inception	0	X	1	√	0	X
11	Flattening	0	X	0	X	1	√
12	Classification	1	√	1	√	1	√
13	Cross Channel Normalization Layer	0	X	2	√	0	X
14	Batch Normalization Layer	0	X	0	X	53	√
15	Addition layer	0	X	0	X	16	√

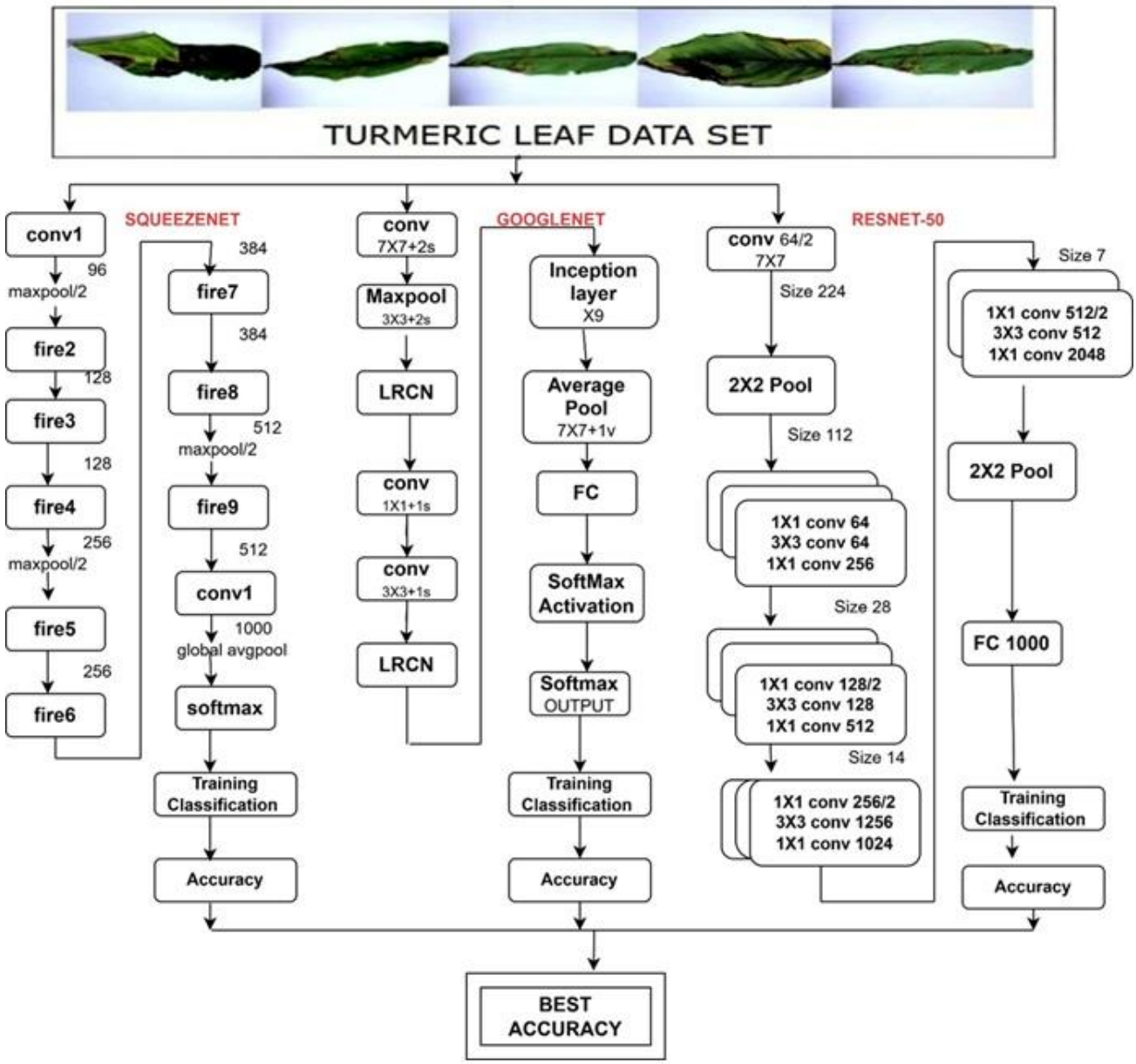

Figure 3. Methodology of training issues to extract best classification accuracy using deep learning architectures

5. ASSUMPTIONS AND MATHEMATICAL NOTATIONS

Although we chose and adjusted the optimal hyperparameters based on a uniform probability distribution function (Epochs and Learning rate with equal intervals), the final findings were accepted using the Poisson distribution to suggest the most effective classifier.

5.1 SGDM

The SGDM algorithm exhibits the ability to oscillate in a direction that is most direct to the desired outcome. This oscillation can be mitigated through the incorporation of a momentum term into the parameter update. SGDM has been modified.

$$\theta_{l+1} = \theta_l - \alpha\nabla E(\theta_l) + \gamma(\theta_l - \theta_{l-1}) \quad (1)$$

The symbol γ represents the contribution to the current iteration from the previous gradient step. Momentum training is a method that may be used to determine this worth. Utilize the SGDM input option under the training options and use SGDM to train a neural network. The initial value for the learning rate should be provided using the initial learning rate training parameter. In addition, different layers and parameters may have different learning rates specified. While using SGDM with momentum, all parameters are learned at the same pace. Through the use of learning rates that change according to parameters and may automatically adapt to the loss function being improved, other optimization techniques seek to enhance network training.

5.2 RMSProp

The RMSProp calculates the moving average by considering the square of each parameter gradient and then averaging them.

$$v_l = \beta_2 v_{l-1} + (1 - \beta_2)[\nabla E(\theta_l)]^2 \quad (2)$$

The moving average decay rate is shown as $\beta 2$. There are three standard decay rates: 0, 9, and 0,999. Averaging the squared gradients over durations of 10, 100 or 1/(1-$\beta 2$) and 1000 parameter updates is required. By using the Squared Gradient Decay Factor (SGDF) training settings, you may pick $\beta 2$. This moving average is used by the RMSProp method to normalize the updates of each parameter separately.

$$\theta_{\iota+1} = \theta_\iota - \frac{\alpha\nabla E(\theta_\iota)}{\sqrt{v_\iota}+\epsilon} \quad (3)$$

where, each part is divided separately. A small constant ε has been added to prevent division by zero and it successfully reduces learning rates for parameters with strong gradients while enhancing learning rates for values with mild gradients. While customization is possible via the Epsilon training option, the default configuration generally functions satisfactorily.

5.3 ADAM

The ADAM employs a parameter update method that bears a resemblance to RMSProp (whereas it is derived from ADAM) and incorporates a momentum factor. The parameter gradients and their squared values are monitored through the utilization of an element-wise moving average.

$$m_l = \beta_1 m_{l-1} + (1 - \beta_1)\nabla E(\theta_l) \quad (4)$$

$$v_l = \beta_2 v_{l-1} + (1 - \beta_2)[\nabla E(\theta_l)]^2 \quad (5)$$

The Squared Gradient Decay Factor (SGDF) and Gradient Decay Factor training parameters, respectively, let you choose the $\beta 1$ and $\beta 2$ decay rates. Moving averages are used by ADAM to change the network's settings as necessary.

$$\theta_{l+1} = \theta_l - \frac{am_l}{\sqrt{v_l}+\epsilon} \quad (6)$$

Parameter adjustments can gain momentum in one direction by employing a moving average of the gradient, provided that the gradients remain equivalent over multiple iterations. A significant portion of the gradient moving average is consumed by noise, leading to a corresponding decrease in parameter updates. One may specify through the utilization of the Epsilon training option. For some issues, a value as large as 1 is more effective than the default value, which is frequently adequate. If you wish to train a neural network using ADAM, the training choice for initial input is ADAM. In addition, the ADAM enhancement as a whole incorporates a method for reversing a bias that manifests itself during training. Utilizing the initial learn rate training parameter, configure the learning rate for every optimization technique. Learning rates that are suitable for a given optimization strategy are also dependent on how that rate influences that strategy. It is also possible to specify distinct learning rates for various layers and parameters.

5.4 Gradient clipping

The training process is characterized by instability and will deviate significantly after a few iterations if the magnitude of the gradient increases exponentially. A training loss that is NaN (Not a Number) or Inf (Infinity) indicates a "gradient explosion". Gradient pruning stabilizes training in the presence of anomalies and at higher learning rates, thereby mitigating the issue of gradient explosion. Gradient clipping enhances the efficiency of network training while preserving the correctness of the learned task. Norm-based gradient clipping adjusts the magnitude of the gradient while preserving its direction. The parameters 'l2norm' and 'global-l2norm' in the Gradient Threshold Method refer to gradient clipping techniques based on norms that partial derivative exceeding the threshold is subjected to value-based gradient cropping, leading to an arbitrary alteration in the gradient's direction. Although the behavior of value-based gradient clipping may appear counterintuitive, the network remains stable even when changes are of a negligible magnitude. The Gradient Threshold Method determines the absolute value of the gradient by employing a value-based gradient reduction strategy.

5.5 L2 regularization

An approach to mitigate overfitting involves incorporating a regularization term into the loss function $E(\theta)$ for the weights Eqs. (1)-(2). The weight decay term is also used in

regularization terms. The form of the loss function including the regularization term is as follows:

$$E_R(\theta) = E(\theta) + \lambda\Omega(w) \quad (7)$$

The regularization function $\Omega(w)$ is denoted by where w signifies the weight vector and λ represents the regularization factor (coefficient).

$$\Omega(w) = \frac{1}{2}w^T w \quad (8)$$

6. DEEP LEARNING ARCHITECTURE

The study mainly focuses on popular SqueezeNet, GoogLeNet, and ResNet-50 deep learning architectures that result in 100 percent training classification accuracy.

6.1 SqueezeNet

SqueezeNet balances precision and simplicity, making it ideal for mobiles and embedded systems with limited resources. Fire modules—specialized convolutional layers that combine 1 × 1 and 3 × 3 filters—set SqueezeNet apart. This combination reduces parameters without sacrificing accuracy, making it ideal for low-resource devices. It may achieve excellent precision with a small fraction of the processing resources of other convolutional neural networks. SqueezeNet uses channel squeezing, a technology breakthrough. By lowering the number of channels in the model's convolutional layers, this method minimizes network computational load while retaining accuracy. SqueezeNet improves efficiency via channel squeezing, fire modules, and deep compression.

6.2 GoogLeNet

The architecture-building inception module is GoogLeNet main invention. Concatenated convolutional layers with varying kernel sizes capture features of different scales in an inception module. This parallelism lets GoogLeNet capture fine-grained details and high-level information, making it great at picture identification. Before applying larger convolutional kernels, each inception module reduces the computational cost by reducing dimensionality with 1 × 1 convolutions.

6.3 ResNet-50

ResNet50 has four main components: convolutional layers, an identification block, and fully connected layers. The identity block and convolutional block process and alter input picture features extracted by the convolutional layers. The final categorization uses fully connected layers. Max pooling layers reduce feature map spatial dimensions while keeping the most important characteristics after convolutional layers. The identification and convolutional blocks comprise ResNet50's core. The identity block simply applies convolutional layers to its input. It then merges input and output. This lets the network learn residual functions that convert input to output. The convolutional block is similar to the identity block, but it has a 1 × 1 convolutional layer to reduce filters before the 3 × 3 layer. Fully connected layers are ResNet50's final component. The final classification falls between these levels. The last fully connected layer output is passed via a SoftMax activation function to generate class probabilities.

In all three architectures a few are common layers such as Input, SoftMax, Global Average Pooling 2D, ReLU, Fully Connected, Max Pooling Layer 2D, Convolution 2D, and Classification layers.

6.4 Input layer

The input layer is the most common in all three deep learning architectures and consists of only one. The principal function of the Input layer is to receive RGB images supplied by artificial input neurons in the input layer of a neural network. This data is processed by artificial neurons in subsequent layers.

6.5 SoftMax layer

SoftMax is operationalized just before the output layer via a neural network layer. The number of nodes in the SoftMax layer should be equivalent to that of the output layer. It is primarily utilized during the classification of training objects and is an activation function. As that of the Input layer, the SoftMax layer count is also only one in all three architectures.

$$S(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \quad (9)$$

where, x is the vector of raw output from the neural network, the e value is 2.718 and i represents the output vector of x with a predicted probability of ith class.

6.6. Global average pooling 2D layer

To perform down sampling, the 2-D global average pooling layer computes the mean of the input's height and width dimensions using the integer factor method. Integer factor down sampling is also known as compression. Integer factor can be achieved by reducing high-frequency signal using a low-pass filter and decimating the filtered signal at M. By using the Finite Impulse Response (FIR) filter is efficient in filtering to decimate Mth output for signal. Therefore, the FIR nth out samples is articulated as:

$$y(n) = \sum_{k=0}^{n} x[nM - k] \,.\, h[k] \quad (10)$$

where, x vector represents the down samples input signals and h vector is impulse response signals of k length.

The layer pools over the spatial dimensions for 2-D picture input comprises four dimensions they are single channel and single observation with two spatial dimensions. Whereas the dimensionality is different for a sequence of images, in addition to the previous four dimensions time steps dimension is added to existing data. The Global Average Pooling 2D layer is common in all three deep learns architectures represented only once in architecture.

6.7 Dropout layer

The Dropout layer plays a crucial role in setting input elements to zero randomly and the probability value is 0.05 by default.

$$E_D = 0.5(t - \sum_{i=1}^{n} \delta_i w_i I_i)^2 \quad (11)$$

Here t represents time, I is the input feature, w is weights and δ is defined as the dropout rate, where the probability equals 1 or zero.

The Dropout layer is presented only once in SqueezeNet and GoogLeNet architectures and this layer is not available in ResNet-50. In ResNet-50 the random values for input values are not assigned as zero.

6.8 Rectified liner unit (ReLU) layer

ReLU layer's each input element is subjected to a threshold operation, which is less than zero to zero. The layer's max value, threshold value, and negative slope values are all larger than or equal to zero in this instance. The ReLU function and its derivative exhibit a monotonic nature. Negative input results in the function returning 0. Positive input, however, causes the function to return x. Thus, the spectrum of the output is infinite to zero.

$$f(x) = max(0, x) \quad (12)$$

All three (SqueezeNet, GoogLeNet and ResNet-50) contain the ReLU layer and it is one of the important layers for classification in deep learning architectures. The SqueezeNet architecture contains 26 layers, then GoogLeNet comprises 57 layers and ResNet-50 has 48 layers.

6.9 Fully connected layer

In the FCL layer, the output size is fixed at two and the input size is initialized to auto. Additionally, the values given for Weight Learn Factor, Weight L2Factor, and Bias Learn Rate Factor are all taken into account as 1, whereas only BiasL2Factor is taken into account as 0. Weights Initializer and Bias Initializer were then utilized to initiate the zeros and Glo rot procedures, respectively. The mathematical formulation denoted as Eq. (13) is utilized to represent a single hidden layer of a feed-forward neural network.

$$\hat{y} = \sigma(xW_1)W_2 \quad (13)$$

where, the hidden layer is linked to the output layer and the input layer is linked to the connected layer. If there are N real-valued features and L hidden units and M output units, then $\hat{y}$ can be articulated as:

$$\hat{y} = \sum_{i=1}^{L} \sigma\big(\langle x, W_{1,i}\rangle\big) . W_{2,i} \quad (14)$$

where, σ is the activation function, x is the feature vector, W_1 and W_2 are weights of the hidden layers. All features and weights are represented by a set of real numbers with a specific range ($x \in \mathbb{R}^{1XN}, W_1 \in \mathbb{R}^{N\,X\,L}, W_2 \in \mathbb{R}^{L\,X\,M}$).

6.10 Max pooling 2D layer

Padding, Pool Size, and Strides comprise the Max Pooling 2Dlayer. To maximize the number of steps taken when training an integer or tuple of three numbers, a window size is utilized. By dividing (3, 3) by 3, the utmost value within a pooling window of size 3 × 3 is obtained. An integer, a pair of integers, or None after that. The worth of bounds. Defines Padding as (0, 0, 0, 0) and Strides as (2, 2), and specifies the distance covered by the pooling window during each pooling step.

$$MaxPooling(X)_{i,j,k} = max_{m,n} X_{i.s_x} + m, j.s_y + n, k \quad (15)$$

where, X denotes the input, (i, j) the output indexes, k the channel index, and (i, j) the horizontal and vertical stride values. The pooling window is established by the filter sizes *fx* and *fy* that are centered at the output index (i, j). The Max Pooling 2D layer is common in all three training networks that were incorporated with different ratios. The SqueezeNet comprises of 3 layers and ResNet-50 contains only in this layer, a 7 × 7 feature map is averaged down to a 1 × 1 dimension. Furthermore, this approach decreases the quantity of trainable parameters to zero and improves the accuracy of the top one by 0.6%. In contrast, GoogLeNet utilizes fourteen layers of MaxPooling to achieve a higher degree of accuracy. The extent of the feature filter in Google Net is determined by the Inception layer, which is responsible for filtering the data (56 × 56 × 1).

6.11 Convolution 2D layer

The Convolution 2D plays a crucial role in all three architectures, where the filter size differs in each architecture. For SqueezeNet, there are twenty-six layers, the filter size is (3, 3), the stride is (2, 2), and the padding value is set to zero. The GoogLeNet is 57, the filter size varies from (3, 3) to (7, 7), the padding value ranges from (0, 0) to (3, 3), and the stride is (2, 2). The ResNet-50 consists of 53 convolution layers, whose filter size ranges from (1, 1) to (14, 14), and padding and stride values are (3, 3) and (2, 2). For instance, the basic characteristics of the Convolution 2D Layer are the number of strides (2, 2), the dilation factor (1, 1), the filter size (3, 3), and the filters (64). Furthermore, we set the Weight2Factor Bias Learn Rate Factor and Weight Learn Rate Factor to 1, the remainder BaseL2Factor to '0', the Base Initializer to zeros, and the Weights Initializer to glorot, among other weight parameters. The convolution filters can be described. The convolution can be defined as

$$Conv = \frac{(W - F + 2P)}{S} \quad (16)$$

6.12 Depth concatenation layer

Inputs with the same height and breadth are concatenated along the channel dimension by a depth concatenation layer. The concatenation layer is available only in SqueezeNet and GoogLeNet.

6.13 Output layer

Only two leaf disease classes, Gemini and Keriting Mosaic are present in the designated output layer for classification, known as the classification layer. Cross entropy ex is used as a loss function and the output size is two.

7. EXPERIMENTATION

This study examines three prominent Deep Learning (DL) architectures: SqueezeNet, GoogLeNet, and Bayesian Deep Learning (BDL)-based ResNet-50. The experiments were conducted using MATLAB 2022A, employing the 'Uniform Probability Distribution' function to optimize hyperparameters, ranging from 30 to 50 epochs and 0.01 to 1 for learning rate, with equal frequency. The specific characteristics of each architecture are delineated in Table 2.

Table 2. Training classification performance analysis of l2norm optimizer on Curcuma longa dataset

S. No.	DL Net	Optimizer	Lowest Accuracy	Epochs	Highest Accuracy	Epochs	Overall Average
1	SQUEEZENET	SGDM	51.35	25	51.35	50	51.35
			51.35	25	51.35	50	51.35
			48.65	25	51.35	30	50.9
			51.35	25	51.35	40	51.35
			51.35	25	51.35	45	51.35
			48.65	25	51.35	30	50.9
		ADAM	97.3	25	100	50	99.55
			48.65	25	51.35	35	49.1
			51.35	25	54.05	50	51.8
			48.65	25	51.35	35	50.9
			48.65	25	51.35	50	50.9
			48.65	40	51.35	25	50
		RMSPROP	48.65	30	100	50	85.58
			48.65	35	100	30	61.26
			48.65	30	70.27	35	53.15
			48.65	50	51.35	30	50
			48.65	25	51.35	50	50.45
			48.65	25	51.35	40	49.55
2	GOOGLENET	SGDM	97.3	40	100	50	99.55
			97.3	25	100	45	99.55
			97.3	35	100	50	99.55
			51.35	25	51.35	45	51.35
			51.35	30	51.35	35	51.35
			51.35	25	51.35	40	51.35
		ADAM	100	25	100	40	100
			100	30	100	45	100
			48.65	25	51.35	45	50.9
			48.65	25	51.35	50	50.9
			51.35	30	51.35	45	51.35
			48.65	30	51.35	40	50.9
		RMSPROP	48.65	25	100	50	59.91
			51.35	30	97.3	25	66.66
			48.65	40	51.35	25	50.45
			48.65	50	51.35	25	49.1
			48.65	30	51.35	35	50.45
			48.65	45	51.35	25	50.9
3	RESNET-50	SGDM	100	25	100	45	100
			100	30	100	50	100
			100	25	100	45	100
			100	30	100	40	100
			100	25	100	40	100
			100	25	100	45	100
		ADAM	94.59	25	100	45	97.74
			89.19	35	100	30	96.84
			89.19	30	100	50	95.45
			89.19	35	100	25	94.19
			89.19	35	100	30	94.59
			89.19	30	100	25	94.29
			89.19	50	100	25	96.39
		RMSPROP	89.19	50	100	35	96.84
			89.19	35	100	25	95.94
			89.19	50	100	30	93.69
			89.19	50	100	25	93.84
			91.89	40	100	25	96.09

7.1 Image data acquisition

Images of plant leaf diseases were acquired from the Customs Dataset, specifically from agricultural fields. The images were subsequently categorized into three distinct categories. The collection portrays Leaf Bloch and Spots, two prevalent leaf diseases that have the potential to affect the plant as mentioned above commodities. The diseases affecting turmeric leaves were selected for the compilation due to their global and Indian recognition, respectively. By default, each

image is converted to a unique JPG file, which adheres to the RGB color space. A dataset has been collected from the field of research Area in Duggirala, Andhra Pradesh. The dataset contains three categories of disease leaf, which have leaf blotch, Colletotrichum leaf spot, and Cercospora leaf spot, respectively. The researcher collected private data from the field of Duggirala, Andhra Pradesh.

8. RESULTS

8.1 SqueezeNet

The present study, carried out on turmeric leaf disease data, was subjected to the SqueezeNet DL model. The training results are concentrated by varying learning rates and ephods. Two categories of results were presented in this section to evaluate the performance of SGDM on SqueezeNet: LR values were held constant while varying epochs, and vice versa. In the experimentation, epochs were considered from 25 to 50, with 5 as an interval, and LR ranges from 0.01 to 0.05 and 1. Along with LR and epochs, the performance is measured on three types of vector norms, such as L2Norm, Global-L2Norm, and absolute value. In general, norms are used to measure the error rate in the training process in DL model.

8.1.1 SGDM

Gradient descent is an iterative optimization technique that seeks the optimal value (minimum or maximum) of an objective function. Finding the model parameters that offer the highest accuracy on both training and test datasets is the main objective of gradient descent. The SDGM is a popular optimizer that plays an important role in achieving 100 percent training accuracy by adjusting bias and weights at each layer. The learning rate, which is a hyperparameter, dictates the extent to which the model is modified whenever the predicted error causes a weight adjustment. Figure 4 describes the loss and gain function of the training data based on the SGDM optimizer.

8.1.2 Impact of learning rate on SGDM

A low learning rate may cause the algorithm to converge slowly in SGDM, whereas a high learning rate may cause it to overshoot the minimum. The performance of SqueezeNet on SGDM based on LR at a constant rate in various epochs is notably shown in Figure 4. SqueezeNet obtained 51.4 percent training accuracy for all epochs and optimizers during LR at 0.02. However, the performance differed between optimizers in varying epochs. However, as compared to the absolute value and global L2 norm, the L2Norm optimizer is high. The SGDM's L2Norm peak performance training accuracy is 51.4, with 48.4 being the lowest.

In Global-L2Norm, where training accuracy is high (51.4) and observed accuracy is low (48.5), the scenario persisted as well. Comparing the Global-L2Norm to the absolute-value optimizer in comparison to the other two in SGDM, the performance of the absolute-value optimizer is minimal. Table 2 shows that the higher epochs, those between 45 and 50 are gaining greater success in SGDM. Overall, SqueezeNet SGDM training accuracy performance is rather poor, and using L2Norm will increase accuracy at all levels of LR and epochs.

8.1.3 Impact of epochs on SGDM

Secondly, to measure the performance of SqueezeNet on SGDM while epochs are constant and varying for different learning rates that vary from 0.01 to 0.05 and 1. The training accuracy on the turmeric dataset is unstable, and it varies according to the learning rate. The training accuracy of the 'Absolute-Value' optimizer showed a peak performance that resulted in 100 percent. Then L2Norm and global-L2Norm performance are almost similar, with 41.2 percent where epochs are at 25 (Table 2). The training accuracy ranged from 48.1 to 55.8 when epochs ranged from 30 to 50, which is lower (Table 2). Thus, it is interpreted that SDGM is very low when epochs are constant. Also, the below results clearly state that LR and Epochs have a lower impact on SDGM in SqueezeNet on the current Turmeric Leaf Diseased dataset.

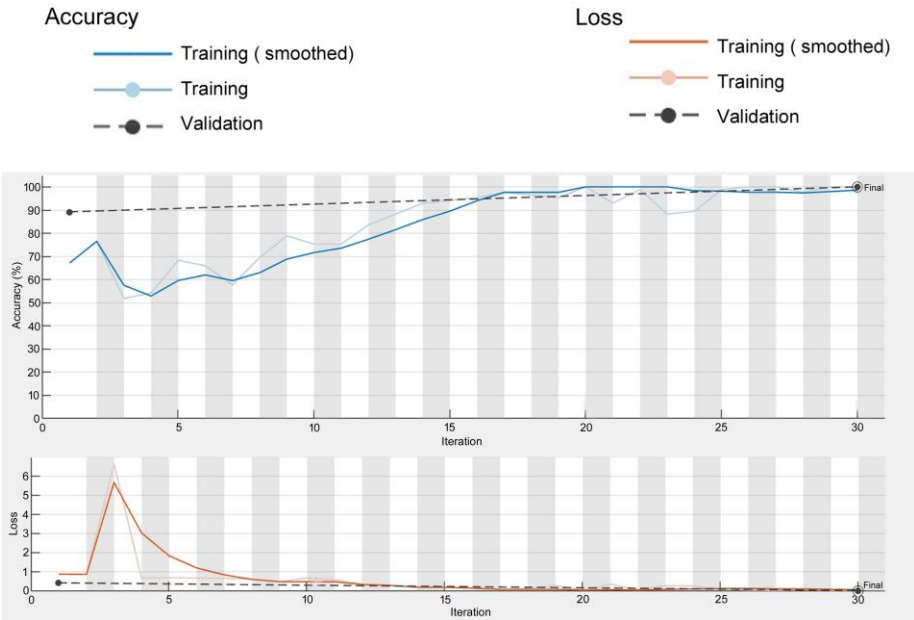

Figure 4. Deep learning SGDM training data gain and loss graphs

8.1.4 ADAM

The ADAM optimizer is incredibly effective when dealing with complex problems involving numerous variables or data, because it is a combination of SGDM and RSMPROP. As per DL literature the ADAM performance is good compared with rest. The presented results show the performance variance by considering three vector norms i.e., L2Norms, Global-L2Norm and Absolute-Value. The section focuses on the ADAM optimizer in SqueezeNet performance analysis on various vectors that yield higher training accuracy for the Turmeric dataset.

8.1.5 Impact of learning rate on ADAM

The performance of ADAM is fairly good (training accuracy = 100 percent) when the LR is at 0.01 for all epochs that range from 25 to 50. However, a gradual decline is observed in accuracy whenever the learning rate reaches from 0.02 to 1. So, it is interpreted that the ADAM results in higher accuracy with a lower learning rate. It was observed that the accuracy resulted differently for each vector norm, where the L2 norm and absolute-value performance were higher than the global L2 norm the global L2 norm for SqueezeNet using the ADAM the ADAM optimizer (Table 2). Even though training accuracy is achieved at a lower rate with a higher learning rate, the absolute value and global L2 norm are stable while using ADAM (SGDM). At the same time, the performance of L2Norm varied from 100 percent to 48.5 percent. From this observation, it is understood that lower learning isn't suitable for attaining higher training accuracy.

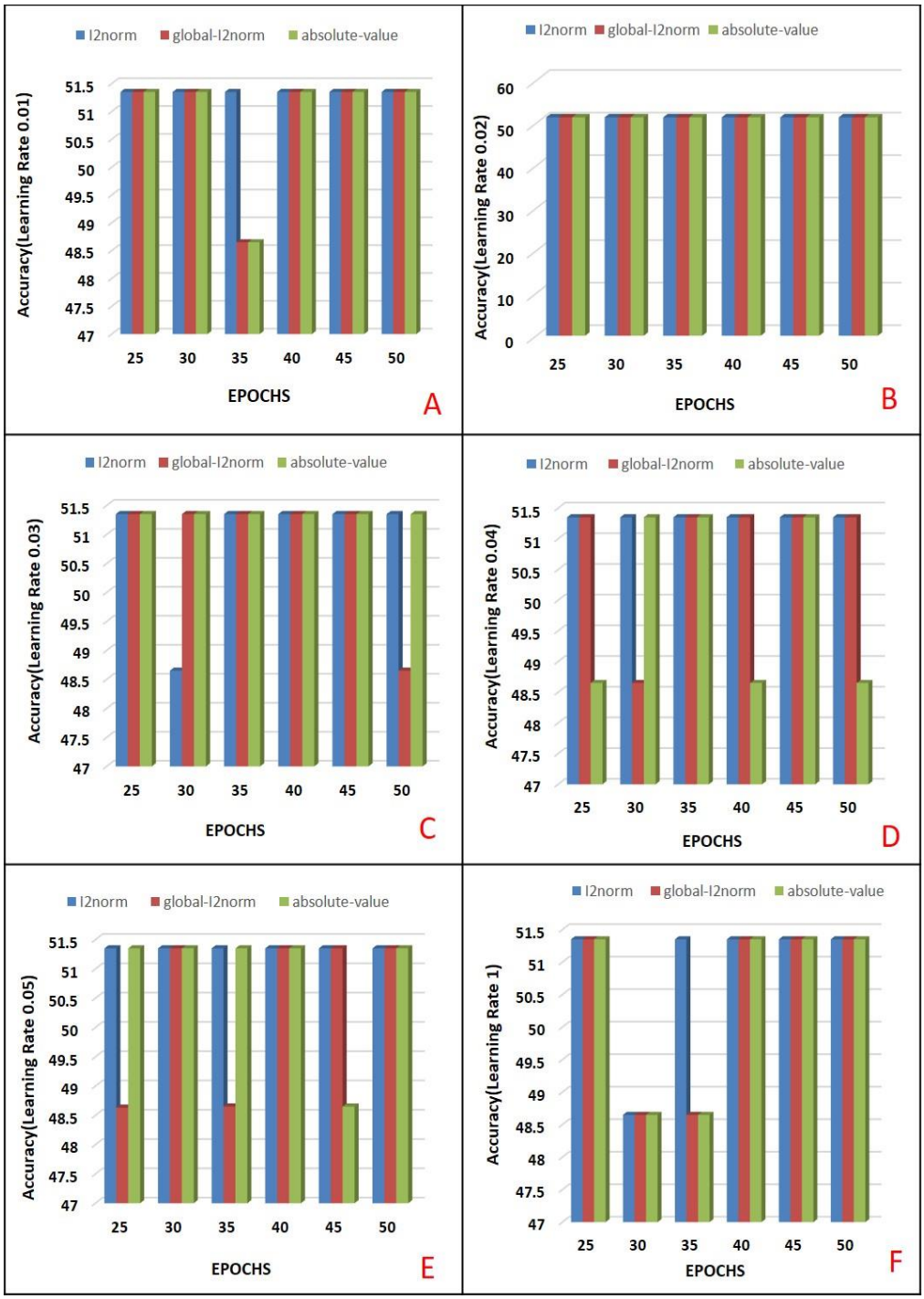

Figure 5. Performance analysis of SqueezeNet on SGDM by making EPOCH as constant

8.1.6 Impact of epochs rate on ADAM

The Epochs have shown lesser impact in improving training accuracy using SqueezeNet (ADAM). For all the six cases the only at lower LR (0.01) has shown greater impact irrespective of Epochs (Figure 5). At every epoch that ranges from 25 to 50 the training accuracy attained 100 percent. In all six cases the L2Norm, Global-L2Norm and Absolute-value have shown a similar behavior except in a few cases, which is negligible accuracy. From the above two observations, it is concluded that ADAM in more concern with the DL rather than Epochs in SqueezeNet. The higher LR may result in overfitting because of lower training accuracies in ADAM (SGDM). The results on SqueezeNet (ADAM) interpret that lower LR is efficient in training the Turmeric diseased dataset.

Table 3. Training classification performance analysis of globall2norm optimizer on Curcuma longa dataset

S.No.	DL Net	Optimizer	Lowest Accuracy	Epochs	Highest Accuracy	Epochs	Over All Average
1	SQUEEZENET	SGDM	48.65	35	51.35	50	50.9
			51.35	25	51.35	45	51.35
			48.65	50	51.35	25	50.9
			48.65	30	51.35	45	50.9
			48.63	25	51.35	50	50.4
			48.65	30	51.35	50	50.4
		ADAM	97.30	35	100	50	99.1
			48.65	30	51.35	50	49.55
			48.65	50	51.35	25	50.9
			48.65	25	51.35	45	50
			48.65	30	51.35	35	50.9
			48.65	50	51.35	30	50.45
			48.65	25	100	50	81.53
		RMSPROP	48.65	25	100	30	58.1
			48.65	25	91.89	45	61.7
			48.65	30	100	45	66.21
			48.65	25	75.68	30	53.15
			48.65	25	51.35	35	50
		SGD	97.3	25	100	45	99.55
			97.3	35	100	50	99.55
			100	25	100	40	100
			51.35	30	51.35	50	51.35
			51.35	25	51.35	45	51.35
			51.35	25	51.35	35	51.35
2	GoogLeNet	ADAM	97.3	25	100	45	99.55
			97.3	25	100	45	99.1
			48.65	25	51.35	45	50
			51.35	30	51.35	40	51.35
			48.65	30	51.35	50	50
			51.35	25	51.35	40	51.35
		RMSPROP	48.65	25	100	45	74.32
			48.65	35	100	50	67.56
			48.65	50	51.35	45	50.9
			48.65	50	51.35	45	50.9
			48.65	25	51.35	50	49.55
			51.35	25	51.35	50	51.35
		SGDM	100	25	100	50	100
			100	25	100	50	100
			100	25	100	50	100
			100	25	100	50	100
			100	25	100	50	100
			100	25	100	50	100
			100	25	100	50	100
3	ResNet-50	ADAM	94.59	30	100	50	99.09
			89.89	40	100	45	95.49
			89.19	50	100	25	95.69
			91.59	35	100	25	95.49
			89.19	40	100	25	94.89
			89.19	45	100	30	95.94
			89.19	45	100	25	93.69
		RMSPROP	89.19	50	100	25	93.69
			89.19	45	100	30	94.59
			89.19	50	100	30	94.14
			85.59	40	98.19	25	92.34

8.1.7 RMSProp

As stated in the literature, RMSProp is one of the popular optimizers for deep learning architectures. The experimentation results confirm that RMSProp performance is much lower than SGDM and ADAM because the RMSProp method is ineffective for mini-batches. when the learning rate

is insufficient. The experimentation was performed by considering the mini-batch size of 128, which is lower, and that resulted in lower accuracy.

8.1.8 Impact of learning rate on RMSProp

The LR impact is very limited on RMSProp in the SGDM optimizer. But RMSProp will achieve higher training accuracy rates (100%) at lower LR and higher epochs. Among the three vector distance methods, Global-L2Norm performance is good, followed by 'Absolute- Value' (Table 3). The overall L2Norm training accuracy performance is lower for LR and at epochs except in the first case (Table 4).

8.1.9 Impact of epochs rate on RMSProp

The change in epochs has shown a significant impact on RMSProp (SqueezeNet). The results show that whenever there is a change in epochs, that reflects on training accuracy. The spikes are high in 'Global-L2Norm and 'Absolute-Value' compared with 'L2Norm'. The lower epochs have more spikes, and at higher epochs, the accuracy is constantly decreased according to LR. The spikes are more observed in 'Absolute- Value' and 'Global L2Norm'. The 'L2Norm' performance training accuracy is lower and linear in most of the cases and shows less accuracy compared with the rest.

8.1.10 SqueezeNet performance analysis

Tables 2-4 show the performance analysis of various distance vector methods on epochs that vary from 25 to 50. The SqueezeNet (SGDM-L2Norm) DL model results in the lowest training accuracy at LR of 0.03 and 1 with 48.65, and at all other LR rates, the training accuracy is moderate, i.e., 51.35. The average training accuracy of SqueezeNet (SGDM-L2Norm) on the Turmeric Leaf dataset is 51.2% for all epochs (25-50). Next, the SqueezeNet (ADAM-L2Norm) DL model resulted in the lowest training accuracy at LR = 0.04, 0.05, and 1 with 48.65 and a reported 100% where LR = 0.01 and epochs = 50. The average training accuracy of SqueezeNet (ADAM-L2Norm) on the Turmeric Leaf dataset is 58.7% for all epochs (25–50). Next, the SqueezeNet (RMSProp-L2Norm) DL model resulted in the lowest training accuracy at LR of 0.01-1 with 48.65 and reported 100% at LR of 0.01-0.02. The average training accuracy of SqueezeNet (RMSProp-L2Norm) on the Turmeric Leaf dataset is 58.33% for all epochs (25-50). Based on the result statistics, it was concluded that the SqueezeNet DL model showed a moderate performance on the Turmeric dataset with three diseased classes.

It is also observed from RMSProp that the higher learning and lower epoch rates resulted in very lower accuracy, which is not suggestive of higher accuracy because of mini-batch (overfitting) issues in SqueezeNet. Compared with the rest the two optimizers presented, the RMSProp training accuracy is much lower due to the failure to address the mini-batch processing problems. It is also observed that RMSProp also attains higher accuracies in a few cases where epochs are 35-50 and LR is 0.01. The overall training performance (mean epochs (25-50) and mean LR (0.01-1)) of SqueezeNet (ADAM) is 7.5% higher than SqueezeNet (SGDM) and 0.037% higher than SqueezeNet (RMSProp). In addition, SqueezeNet (RMSProp) training accuracy is 7.13% higher than SqueezeNet (SGDM) when the 'L2Norm' vector distance method is applied. The overall training performance (Mean Epochs (25-50) and Mean LR (0.01-1)) of SqueezeNet (ADAM) is 7.67% higher than SqueezeNet (SGDM) and 3.29% lower than SqueezeNet (RMSProp). In addition, SqueezeNet (RMSProp) training accuracy is 10.97% higher than SqueezeNet (SGDM) when the 'Global_L2Norm' vector distance method is applied.

The overall training performance (Mean Epochs (25-50) and Mean LR (0.01-1)) of SqueezeNet (ADAM) is 7.95% higher than SqueezeNet (SGDM) and 7.65% lower than SqueezeNet (RMSProp). In addition, SqueezeNet (RMSProp) training accuracy is 15.61% higher than SqueezeNet (SGDM) when subjected to the 'Absolute' vector distance method.

8.2 GoogLeNet

8.2.1 SGDM

Gradient descent is an iterative optimization technique that seeks the optimal value (minimum or maximum) of an objective function. Finding the model parameters that offer the highest accuracy on both training and test datasets is the main objective of gradient descent. The SDGM is a popular optimizer that plays an important role in achieving 100 percent training accuracy by adjusting bias and weights at each layer.

8.2.2 Impact of learning rate on SGDM

In SGDM, a low learning rate can result in slow convergence of the algorithm, while a high learning rate can cause it to exceed the minimum value. The performance of GoogLeNet on (SGDM) based on the Learning Rate (LR) at a consistent rate across different epochs. According to Table 4, GoogLeNet achieved a training accuracy of 100 percent for smaller epochs and optimizers when the learning rate (LR) was set to 1. Nevertheless, there was a variation in the performance of the optimizers in different vectors. However, as compared to the absolute value and global L2 norm, the L2Norm optimizer is elevated. The SGDM's L2Norm peak performance training accuracy is 100, while the lowest accuracy recorded is 50. The scenario of persistently low observed accuracy (51.5) despite high training accuracy (100) continued in Global-L2Norm. Figure 6 shows the varied epochs performance under the constant learning rate.

Contrasting the Global-L2Norm optimizer with the absolute-value optimizer. The absolute-value optimizer has the lowest performance among the other two optimizers in SGDM. The epochs between 45 and 50 exhibit higher levels of success in SGDM. In general, the training accuracy performance of Google Net SGDM is quite unsatisfactory. However, by employing Global-L2Norm, accuracy can be enhanced across all LR and epoch levels.

8.2.3 Impact of epochs rate on SGDM

Second, to assess Google Net performance on SGDM while keeping epochs constant and varied for different learning rates ranging from 0.01 to 0.05 and 1. The training accuracy on the turmeric dataset is inconsistent and varies with learning pace. The training accuracy of the 'Global-L2Norm ' optimizer has reached a high of 100 percent. When Epochs are set to 25, L2Norm and Absolute-value performance are nearly identical (Table 3). When the epochs ranged from 30 to 50, the training accuracy was 50 to 55.1. As a result, it is assumed that SDGM is very low when epochs remain constant. Also, the results show that LR and Epochs have a reduced influence on SDGM in GoogLeNet on the current Turmeric leaf sick dataset.

Table 4. Training classification performance analysis of absolute value optimizer on Curcuma longa dataset

S.No.	DL Net	Optimizer	Lowest Accuracy	Epochs	Highest Accuracy	Epochs	Average
1	SQUEEZENET	SGDM	48.65	35	51.35	50	50.9
			51.35	25	51.35	40	51.35
			51.35	30	51.35	45	51.35
			48.65	25	51.35	45	50
			48.65	30	51.35	50	50.45
			48.65	35	51.35	25	50
		ADAM	100	25	100	45	100
			48.65	25	51.35	40	50.45
			48.65	30	51.35	50	50.45
			48.65	25	51.35	45	50.45
			48.65	30	51.35	50	50.45
			48.65	35	51.35	25	50
		RMSPROP	48.65	30	100	45	85.12
			51.35	50	100	30	82.42
			51.35	25	100	50	73.42
			48.65	50	100	30	58.1
			48.65	25	51.35	40	49.1
			48.65	30	51.35	45	49.55
2	GoogLeNet	SGDM	100	25	100	50	100
			100	30	100	45	100
			51.35	25	51.35	40	51.35
			51.35	30	51.35	50	51.35
		ADAM	48.65	25	51.35	40	50.9
			51.35	35	51.35	50	51.35
			97.3	35	100	25	99.55
			94.59	25	100	40	98.19
			48.65	25	51.35	45	50
			48.65	30	51.35	50	50.9
			48.65	30	51.35	45	50
			51.35	35	51.35	50	51.35
		RMSPROP	48.65	35	51.35	45	50.9
			48.65	30	91.89	40	68.91
			48.65	25	51.35	40	50.45
			48.65	25	51.35	35	49.55
			48.65	30	51.35	45	50
			48.65	25	51.35	45	50
3	ResNet-50	SGDM	100	30	100	45	100
			100	25	100	35	100
			100	35	100	25	100
			100	25	100	45	100
			100	30	100	45	100
			100	25	100	50	100
		ADAM	100	30	100	25	100
			91.89	30	100	25	98.64
			89.19	40	100	30	94.74
			89.19	35	100	25	94.14
			89.19	35	100	30	93.69
			89.19	35	100	25	92.79
		RMSPROP	91.89	25	100	40	97.29
			91.89	35	100	30	95.49
			91.89	50	100	25	95.04
			91.89	35	100	25	96.39
			91.89	35	100	25	94.74
			89.19	35	98.19	30	92.49

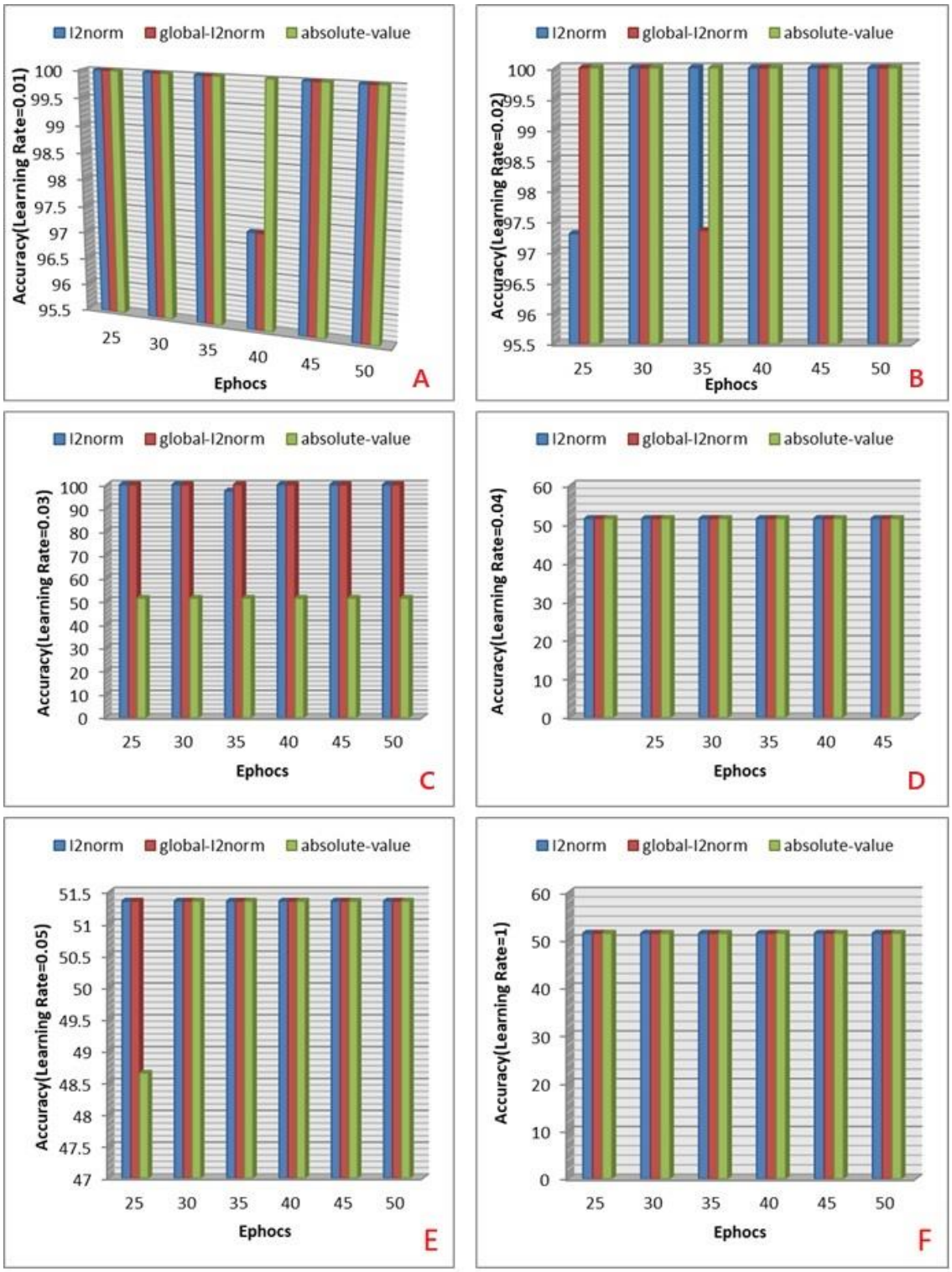

Figure 6. Performance analysis of GoogLeNet on SGDM by making LR as constant

8.2.4 ADAM

The ADAM is one finest optimizer, where the training accuracy is achieved 100%. the gain and loss training accuracy shows the performance measure of the Turmeric Dataset using ADAM.

8.2.5 Impact of learning rate on ADAM

ADAM exhibits a respectable level of performance (training accuracy = 100 percent) across all epochs in the range of 25 to 50 when the LR is set to 0.01. However, an accuracy decline is observed gradually as the learning rate increases from 0.02 to 1. Thus, it was deduced that ADAM yields greater precision at a reduced learning rate. Additionally, it was noted that the accuracy outcomes varied among the vector norms, with absolute-value and global-l2norm performance surpassing that of L2Norm for GoogLeNet when the ADAM optimizer was utilized (Table 3). Even though training accuracy decreases as the learning rate increases, the performance of the Global-L2Norm and absolute value remain consistent when ADAM is utilized. Global-l2norm performance, on the other hand, varied between 51.5 and 100 percent. It is evident from this observation that lower learning algorithms are not optimal for achieving higher training accuracy. The performance of varied epochs with constant learning rate is shown in Figure 7.

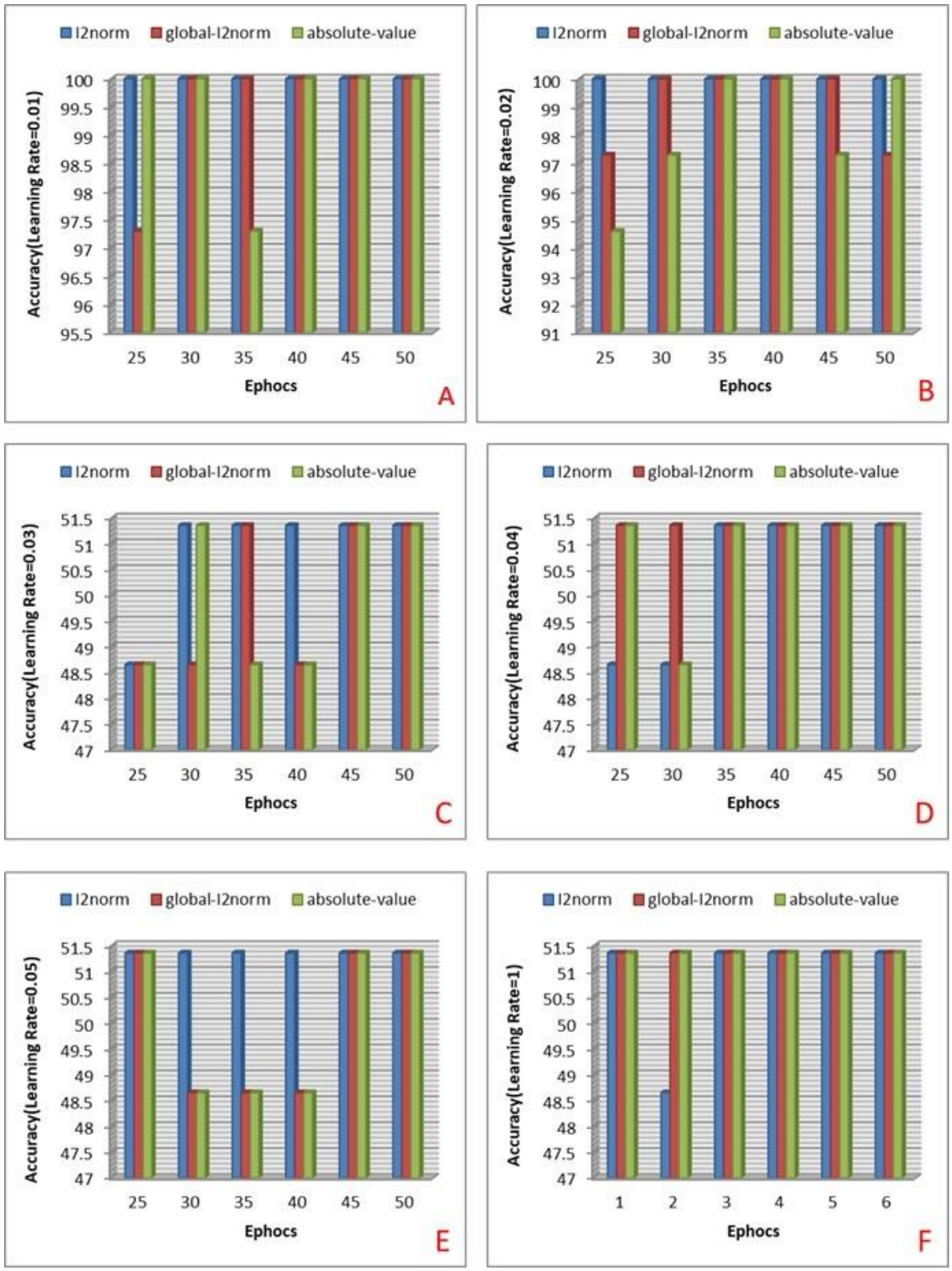

Figure 7. Performance analysis of GoogLeNet on ADAM by making LR as constant

8.2.6 Impact of epochs rate on ADAM

ADAM exhibits a respectable level of performance (training accuracy = 100 percent) across all epochs in the range of 25 to 50 when the LR is set to 0.01. However, an accuracy decline is observed gradually as the learning rate increases from 0.02 to 1. Thus, it was deduced that ADAM yields greater precision at a reduced learning rate. Additionally, it was noted that the accuracy outcomes varied among the vector norms, with absolute-value and global-l2norm performance surpassing that of L2Norm for GoogLeNet when the ADAM optimizer was utilized.

Even though training accuracy decreases as the learning rate increases, the performance of the Global-L2Norm and absolute value remain consistent when ADAM is utilized. Global-l2norm performance, on the other hand, varied between 51.5 and 100 percent. It is evident from this observation that lower learning algorithms are not optimal for achieving higher training accuracy.

8.2.7 RMSProp

According to the literature, one of the common optimizers for deep learning systems is RMSProp. Because the RMSProp method is unsuccessful for mini batches, its performance is substantially lower than that of SGDM and ADAM.

8.2.8 Impact of learning rate on RMSProp

The LR impact is very limited on RMSProp in the SGDM optimizer. But RMSProp will achieve higher training accuracy rates (97.3%), at lower LR and higher Epochs (Table 4). Among the three vector distance methods, Global-L2Norm performance is Average, followed by 'Absolute-value' (Table 4). The overall global-L2Norm training accuracy performance is lower for LR and at Epochs except in the third.

8.2.9 Impact of epochs rate on RMSProp

The change in Epochs has shown a significant impact on RMSProp (GoogLeNet). The results show that whenever there is a change in Epochs and that reflects on training accuracy. The spikes are high in 'Global-L2Norm and 'L2Norm' compared with 'absolute value (Table 4). The lower epochs have more spikes and at higher epochs, the accuracy is constantly decreased according to LR.

The Spikes are more observed in 'Absolute-value' and 'Global-L2Norm'. The 'L2Norm' performance training accuracy is lower and linear in most of the cases and shows less accuracy compared with the rest.

8.2.10 GoogLeNet performance analysis

The performance analysis of various distance vector methods on epochs that vary from 25 to 50. The GoogLeNet (SGDM-L2Norm) DL model results in the lowest training accuracy as 51.35% at LR is 0.03-1and the rest of the all LR rates the training accuracy is achieved high accuracy i.e., 100%. The average training accuracy of GoogLeNet (SGDM-L2Norm) on the Turmeric leaf dataset is 75.45% for all epochs (25-50). Next, the GoogLeNet (ADAM-L2Norm) DL model resulted in the lowest training accuracy at LR of 0.03-1with 48.65% and reported 100% where LR is 0.01-0.02 and Epochs are between 40-45. GoogLeNet (ADAM-L2Norm) has 67.34% training accuracy on the Turmeric leaf dataset for all epochs (25-50). GoogLeNet (RMSProp-L2Norm) DL model has the lowest training accuracy at 48.65% (except at LR = 0.02) and 100% at LR = 0.01. GoogLeNet (RMSProp-L2Norm) has 54.57% training accuracy on the Turmeric leaf dataset for all epochs (25-50). GoogLeNet DL performed high to moderate on the Turmeric dataset, according to the result statistics. Total training performance (Mean Epochs (25-50) and Mean LR (0.01-1)) of GoogLeNet (SGDM) is 8.10% higher than ADAM and 12.76% higher than RMSProp.

The 'L2Norm' vector distance method improves GoogLeNet (SGDM) training accuracy by 20.87% over RMSProp. GoogLeNet (SGDM) has 8.63% training performance (Mean Epochs (25-50) and Mean LR (0.01-1). greater than GoogLeNet (ADAM) and 9.46% higher than RMSProp. The 'Global_L2Norm' vector distance approach improves GoogLeNet (SGDM) training accuracy by 18.09% over RMSProp. GoogLeNet (SGDM) outperforms GoogLeNet (RMSProp) by 0.82% and 13.36% in training performance (Mean Epochs (25-50) and Mean LR (0.01-1). The 'Absolute' vector distance method improves GoogLeNet (SGDM) training accuracy by 14.19% over RMSProp.

8.3 Resnet-50

8.3.1 SGDM

Gradient descent is an iterative optimization method that finds the best value (the lowest or highest point) for a goal function. The basic goal of gradient descent is to find the model parameters that give the best results on both the training and test datasets. By changing bias and weights at each layer, the SDGM is a well-known optimizer that is a key part of getting 100% training accuracy.

8.3.2 Impact of learning rate on SGDM

In SGDM, a low learning rate may lead the algorithm to converge slowly, whereas a high learning rate may cause it to overshoot the minimum. Table 3 depicts the performance of ResNet-50 on SGDM based on LR at a rate throughout many epochs. ResNet-50 got 100 percent training accuracy for all epochs and optimizers for each LR. The performance of optimizers varied. However, the absolute value, global L2 norm, and L2Norm optimizers are all high. The SGDM's peak performance training accuracy is 100. In SGDM; a low learning rate may lead the algorithm to converge slowly, whereas a high learning rate may cause it to overshoot the minimum.

Table 3 depicts the performance of ResNet-50 on SGDM based on LR at a constant rate throughout many epochs. According to Table 3, ResNet-50 got 100 percent training accuracy for all epochs and optimizers for each LR. however, the performance of optimizers varied the absolute value, global L2 norm, and L2Norm optimizers are all high. The SGDM's peak performance training accuracy is 100. Figure 8 shows the performance of constant Learning Rate of ADAM.

8.3.3 Impact of epochs rate on SGDM

To assess the performance of ResNet-50 on SGDM with constant and variable epochs for different learning rates ranging from 0.01 to 0.05 and 1. The training accuracy on the turmeric dataset is consistent and steady concerning the learning rate.

The training accuracy of the 'Absolute-Value','global-l2norm', and'l2norm'optimizer reached a high of 100 percent. The performance of L2Nor, Global-L2Norm, and l2norm is virtually the same to 100 percent where Epochs are 25-50.

When the epochs ranged from 30 to 50, the training accuracy was 100, which is higher (Table 4). As a result, it is assumed that SDGM is quite high when epochs remain constant. Also, the results show that LR and Epochs have a greater influence on SDGM in ResNet-50 on the current Turmeric leaf-infected data.

8.3.4 ADAM

It is a combination of SGDM and RSMPROP, which makes the ADAM optimizer very good at solving hard problems with lots of factors or data. It has been said in DL writings that ADAM works better than the rest. This part is all about the ADAM optimizer in ResNet-50 and how well it works with different vectors to make training more accurate for the Turmeric dataset.

8.3.5 Impact of learning rate on ADAM

The performance of ADAM is fairly good (Training accuracy = 100 percent for absolute value and global-l2norm) when the LR is at 0.01 for all epochs that range from 25 to 50. But a gradual decline is observed in accuracy whenever the Learning Rate reaches from 0.03 to 1. So, it is interpreted that the ADAM results in higher accuracy with a lower learning rate. It also observed that the accuracy resulted differently for each vector norms, where global-L2norm and absolute-value performance were higher than L2Norm for ResNet-50 using the ADAM optimizer (Table 2). Even though training accuracy resulted at a lower rate with a higher learning rate,

the performance of Absolute-value and Global-L2 Norm resulted is stable while using ADAM. At the same time, the performance of global-l2Norm varied from 100 percent to 89.19 percent. From this observation, it is understood that higher learning rates aren't suitable for attaining higher train accuracy.

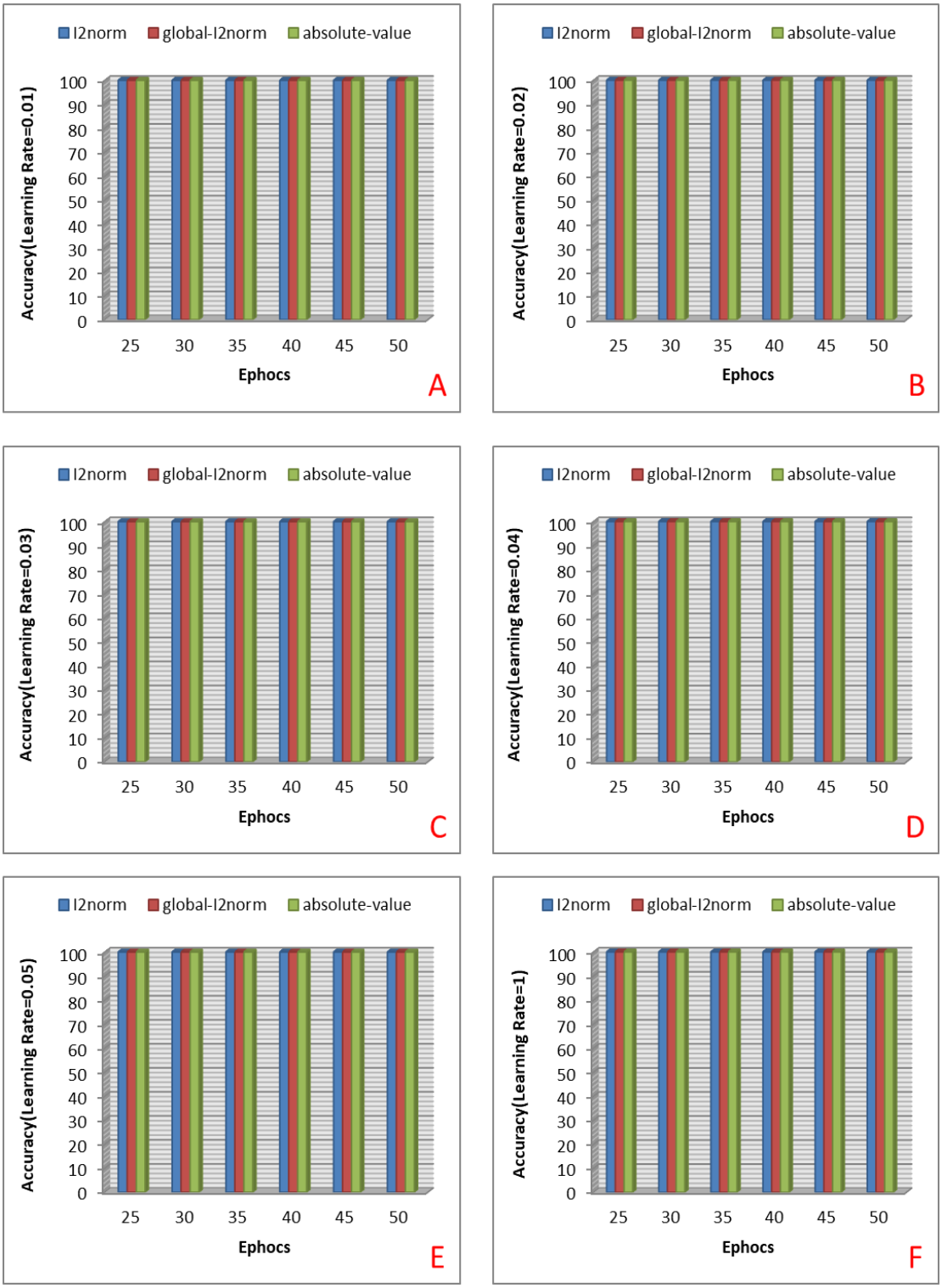

Figure 8. Performance analysis of ResNet-50 on SGDM by making LR as constant

8.3.6 Impact of epochs rate on ADAM

A higher influence of Epochs on ResNet-50 (ADAM) training accuracy has been demonstrated. Regardless of Epoch, Table 2 shows that the instance with the lowest LR (0.01) has the most effect. Between epochs 25 and 50, the training accuracy was perfect every time. After being compared against Global-L2Norm and Absolute-value, L2Norm was shown to be significantly less accurate in all six scenarios.

In light of these two facts, we may conclude that in ResNet-50, ADAM is more concerned with the DL than the Epochs. With lesser training accuracies in ADAM, the larger LR may lead to overfitting. The results of ResNet-50 (ADAM) on the Turmeric diseased dataset show that lower LR is effective in training.

8.3.7 RMSProp

RMSProp is a well-known way to improve deep learning systems. It was proven through experiments that RMSProp doesn't work as well as SGDM and ADAM because it's not good for small epochs. When the experiment was done, the mini-batch size of 128 was used.

8.3.8 Impact of learning rate on RMSProp

When using the SGDM optimizer, the effect of LR on RMSProp is negligible. However, RMSProp will become closer to 100% training accuracy for lower LR and longer Epochs (Table 4). Good results are shown with the 'Absolute-value' approach, followed by the 'l2-norm' and the 'global-l2norm', among the three vector distance methods. Except for the first scenario, LR and Epochs result in worse absolute-value training accuracy performance.

8.3.9 Impact of epochs rate on RMSProp

Epochs have been proven to significantly affect RMSProp (ResNet-50). The findings demonstrate that the accuracy of training is affected anytime there is a shift in Epochs. While 'Global-L2Norm' and 'L2Norm' show lower spikes, 'Absolute-Value' has high performance. (Table 4).

There are more spikes in the earlier epochs, and the accuracy steadily declines in later ones, as shown by Learning Rate. The Spikes are more noted in 'Absolute-value'. In most instances, the 'L2Norm' performance training accuracy is lower and linear, and it has proven less accurate than the others.

8.3.10 Resnet-50 performance analysis

Table 4 shows the performance analysis of various distance vector methods on epochs that vary from 25 to 50. The ResNet-50 (SGDM-L2Norm) DL model achieved high accuracy i.e., 100% irrespective of LR and Epochs. The average training accuracy of ResNet-50 (SGDM-L2Norm) on the Turmeric leaf dataset is 100% for all epochs (25-50). Next, the ResNet-50 (ADAM-L2Norm) DL model resulted in the lowest training accuracy at LR is 0.02-1with 89.19% (Epochs are 30-35) and reported 100% where LR is 0.01-1 and Epochs are between 25-50. The average training accuracy of ResNet-50 (ADAM-L2Norm) on the Turmeric leaf dataset is 95.44% for all epochs (25-50). Then the ResNet-50 (RMSProp-L2Norm) DL model resulted in the lowest training accuracy of 89.19% for all LR (except at LR is 1) and reported 100%, where, LR of 0.01-1 for all epochs. The average training accuracy of ResNet-50 (RMSProp-L2Norm) on the Turmeric leaf dataset is 95.46% for all epochs (25-50). Based on the result statistics, it was concluded that the ResNet-50 DL model gave a high performance on the Turmeric dataset.

Total training performance (Mean Epochs (25-50) and Mean LR (0.01-1)) of ResNet-50 (SGDM) is 4.55% higher than ADAM and 4.53% higher than RMSProp. The 'L2Norm' vector distance approach improves ResNet-50 (RMSProp) training accuracy by 0.01% over ResNet-50 (ADAM). The total training performance (Mean Epochs (25-50) and Mean LR (0.01-1)) of ResNet-50 (SGDM) is 3.22% higher than ADAM and 5.93% higher than RMSProp. The 'Global_L2Norm' vector distance approach improves ResNet-50 (ADAM) training accuracy by 2.71% over RMSProp. The ResNet-50 (SGDM) outperforms the ResNet-50 (RMSProp) by 4.33% and 4.76%, respectively, in training performance (Mean Epochs (25-50) and Mean LR (0.01-1). The 'Absolute' vector distance method improves ResNet-50 (ADAM) training accuracy by 14.19% over RMSProp.

9. PERFORMANCE EVALUATION METRICS

The objective of this study was to find the best classification training method for smaller datasets such as Duggirala Curcuma longa (Turmeric), which is a one-of-a-kind plant. Data science literature normally covers many attributes and provides concise explanations, hence these studies are rare. This section compares three architectures—SqueezeNet, GoogLeNet, and ResNet-50—based on three optimizers (SDGM, ADAM, RMSProp) for turmeric leaf disease detection. All three deep learning architectures are applied to Duggirala variant turmeric data's three damaged leaf classes (Leaf Blotch, Colletotrichum leaf spot, and Cercospora leaf spot). Nine comparison evaluations using LR and vector distance measuring methods were provided to find the best deep learning network for higher training classification accuracy. Superimposed line plots show all three network analysis types.

The results show that SqueezeNet balances precision and computational resources well. Traditional CNNs like GoogLeNet and ResNet-50 are accurate but require a lot of computational power to train and deploy. This disqualifies them for embedded and mobile systems. As a feature extractor, SqueezeNet may allow other machine learning pipelines to access its learned features. GoogLeNet deep architecture, accurate network uses many inception modules. This lowers processing and overfitting. The researchers fixed the vanishing gradient problem during training with intermediate layer auxiliary classifiers. Before using larger convolutions, these layers use 1 × 1 convolutions to reduce input channels. This drastically reduces the computational load while maintaining network capacity. Deep neural network development increasingly uses bottleneck layers. ResNet50 has many convolutional layers, batch normalization, and ReLU activation. Most of these layers extract edges, textures, and forms from the image. ResNet-50 is particularly advantageous for tasks that need very complex architectures, such as the identification of diseases in leaves and the segmentation of images. The performance of the GoogLeNet, SqueezeNet and ResNet-50 is showing lower accuracy on three classes of turmeric diseased leaf data while subjected to the RMSProp optimizer at all epochs. SqueezeNet, GoogLeNet and ResNet-50 are giving peak performance in training the turmeric data when ADAM is applied. The average training accuracy is observed in the SGDM model based on LR and Epochs as mentioned in Tables 2-4. The current study solely emphasizes training accuracy and does not consider validation accuracy concerning the proposed dataset. While the training accuracy reaches a perfect score of 100%, there is uncertainty regarding the ability of the test accuracy to reach a high level of 99%. The primary objective is to identify diseases in the leaf dataset; hence, validation plays a critical role in assessing its performance.

9.1 Impact of learning rate

Learning Rate (LR) affects training parameters. It shows the varies of three performance types. Figure 9 displays the mean epochs (25-50) and LR (0.01) performance. Using the 'L2Norm' vector measurement method, ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) by 48.2% and GoogLeNet by 0.045%. GoogLeNet (SGDM) has 48.2% training accuracy compared to SqueezeNet. In ADAM usage, ResNet-50 (ADAM) has 0.045% higher training accuracy than SqueezeNet and 2.26% lower than GoogLeNet. The

performance analysis of ResNet-50 (RMSProp) is 10.81% greater than SqueezeNet and 36.48% higher than GoogLeNet. Using the 'Global_L2Norm' vector measurement method, ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) by 49.1% and GoogLeNet by 0.045%. ResNet-50 (ADAM) has 0.9% higher training accuracy than SqueezeNet and 0.045% higher than GoogLeNet for ADAM use. Finally, ResNet-50 (RMSProp) outperforms SqueezeNet (14.41%) and GoogLeNet (21.62%). ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) by 49.1% and equals GoogLeNet (SGDM) using the 'Absolute' vector measurement approach. ResNet-50 (ADAM) has a training accuracy that is equal to SqueezeNet and 0.045% greater than GoogLeNet when using ADAM. Finally, ResNet-50 (RMSProp) outperforms SqueezeNet (12.17%) and GoogLeNet (46.39%).

Figure 10 shows the mean epochs and LR (0.02) average performance. Using the 'L2Norm' vector measuring technique, ResNet-50 (SGDM) surpasses SqueezeNet (SGDM) by 48.65 and GoogLeNet (SGDM) by 0.045%. The training accuracy of GoogLeNet (SGDM) is 48.25%, somewhat higher than SqueezeNet. ResNet-50's ADAM training accuracy is 47.74% higher than SqueezeNet's and 3.16% lower than GoogLeNet's. Finally, ResNet-50 (RMSProp) surpasses SqueezeNet and GoogLeNet by 35.58% and 30.018% respectively. The 'Global_L2Norm' vector measuring technique gives ResNet-50 (SGDM) a 48.65% higher performance than SqueezeNet and 0.045% higher than GoogLeNet. ADAM training accuracy for ResNet-50 (ADAM) is 49.54% higher than SqueezeNet and 0.01% lower than GoogLeNet. ResNet-50 (RMSProp) outperforms SqueezeNet (35.59%) and GoogLeNet (26.13%). ResNet-50 performance is analyzed using absolute vector measurements. ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) by 48.65% and matches GoogLeNet. The training accuracy of ResNet-50 (ADAM) utilizing ADAM is 48.19% higher than SqueezeNet and 0.045% higher than GoogLeNet. Finally, ResNet-50 (RMSProp) outperforms SqueezeNet (13-07%) and GoogLeNet (26.58%).

Figure 11 shows the average epochs (25-50) and learning rate (0.03) performance. ResNet-50 (SGDM) performance is analyzed using L2Norm vector measuring. The performance of ResNet-50 (SGDM) is 49.1% higher than SqueezeNet and 0.045% higher than GoogLeNet. GoogLeNet (SGDM) has 48.65% training accuracy, higher than SqueezeNet. The training accuracy of ResNet-50 (ADAM) utilizing ADAM is 43.24% higher than SqueezeNet and 44.14% higher than GoogLeNet. The Final performance evaluation of ResNet-50 (RMSProp) is 42.79% lower than SqueezeNet and 45.49% higher than GoogLeNet.ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) by 49.1% using the 'Global_L2Norm' vector measurement technique and is comparable to GoogLeNet.

The training accuracy of ResNet-50 (ADAM) utilizing ADAM is 44.59% higher than SqueezeNet and 45.49% higher than GoogLeNet. Finally, ResNet-50-RMSProp outperforms SqueezeNet by 31.99% and GoogLeNet by 42.79%. ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) and GoogLeNet (SGDM) utilizing the 'Absolute' vector measuring technique by 48.65%. The training accuracy of ResNet-50 (ADAM) utilizing ADAM is 44.29% higher than SqueezeNet and 44.74% higher than GoogLeNet. Performance research shows ResNet-50 (RMSProp) outperforms SqueezeNet (21.62%) and GoogLeNet (44.59%).

Figure 12 shows the average epochs (25-50) and learning rate (0.04) performance. ResNet-50 (SGDM) outperforms SqueezeNet and GoogLeNet by 48.65% using L2Norm vector measurement. SqueezeNet and GoogLeNet (SGDM) have similar training accuracy. ResNet-50 (ADAM) outperforms SqueezeNet and GoogLeNet in ADAM training accuracy by 43.29%. The Final performance evaluation of ResNet-50 (RMSProp) is 43.69% lower than SqueezeNet and 44.59% higher than GoogLeNet. ResNet-50 (SGDM) performance is tested using 'Global_L2Norm' vector measurement. It's 49.1% higher than SqueezeNet (SGDM) and 48.65% higher than GoogLeNet. ResNet-50's training accuracy using ADAM is 45.69% higher than SqueezeNet's and 44.34% higher than GoogLeNet's. In conclusion, ResNet-50 (RMSProp) outperforms SqueezeNet (RMSProp) by 28.38% and GoogLeNet by 43.69%. The 'Absolute' vector measuring method shows that ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) by 50% and GoogLeNet (SGDM) by 48.65%. ResNet-50's training accuracy using ADAM is 43.69% higher than SqueezeNet's and 43.24% higher than GoogLeNet's. Performance investigation shows ResNet-50 (RMSProp) surpasses SqueezeNet (RMSProp) by 38.29% and GoogLeNet by 46.84%.

Figure 13 shows the mean epochs (25-50) and LR (0.05) performance. ResNet-50 (SGDM) outperforms SqueezeNet and GoogLeNet by 48.65% using the 'L2Norm' vector measurement method. Additionally, GoogLeNet (SGDM) has the same training accuracy as SqueezeNet. ResNet-50 (ADAM) has 43.69% higher training accuracy than SqueezeNet and 43.24% lower than GoogLeNet while using ADAM. In conclusion, ResNet-50 (RMSProp) performs 43.39% lower than SqueezeNet and 43.39% higher than GoogLeNet. The 'Global_L2Norm' vector measuring method is used to evaluate ResNet-50 (SGDM), which outperforms SqueezeNet and GoogLeNet by 49.6% and 48.65%, respectively. When ADAM is used, ResNet-50 (ADAM) has 44.59% higher training accuracy than SqueezeNet and 45.49% higher than GoogLeNet. Finally, ResNet-50 (RMSProp) outperforms SqueezeNet (43.39%) and GoogLeNet by the same margin. ResNet-50 (SGDM) performance investigation shows that the 'Absolute' vector measurement approach is 49.1% higher than GoogLeNet and 49.55% higher than SqueezeNet. When ADAM is used, ResNet-50 (ADAM) has 43.24% higher training accuracy than SqueezeNet and 43.69% higher than GoogLeNet. In conclusion, ResNet-50 (RMSProp) outperforms SqueezeNet (45.64%) and GoogLeNet (44.74%).

Figure 14 shows the mean epochs (25-50) and LR (1) performance. ResNet-50 (SGDM) outperforms GoogLeNet (SGDM) by 48.65% and SqueezeNet by 49.1% using the 'L2Norm' vector measurement method. Compared to SqueezeNet, GoogLeNet (SGDM) has 0.045% training accuracy. When ADAM is used, ResNet-50 (ADAM) has 43.29% higher training accuracy than SqueezeNet and 43.39% higher than GoogLeNet. Finally, ResNet-50 (RMSProp) outperforms SqueezeNet (46.54%) and GoogLeNet (45.19%). The 'Global_L2Norm' vector measurement method shows that ResNet-50 (SGDM) outperforms GoogLeNet (SGDM) by 48.65% and SqueezeNet by 49.6%. ResNet-50 (ADAM) has 43.54% higher training accuracy than SqueezeNet and GoogLeNet when ADAM is implemented. The performance of ResNet-50 (RMSProp) is 42.34% higher than SqueezeNet and 40.99% higher than GoogLeNet. The 'Absolute' vector measuring approach gives ResNet-50 (SGDM) a performance rating 50% higher than SqueezeNet and 48.65% higher than GoogLeNet. Using ADAM, ResNet-50 (ADAM) improves

training accuracy by 42.79% over SqueezeNet and 41.44% over GoogLeNet.ResNet-50 (RMSProp) outperforms SqueezeNet (42.94%) and GoogLeNet.

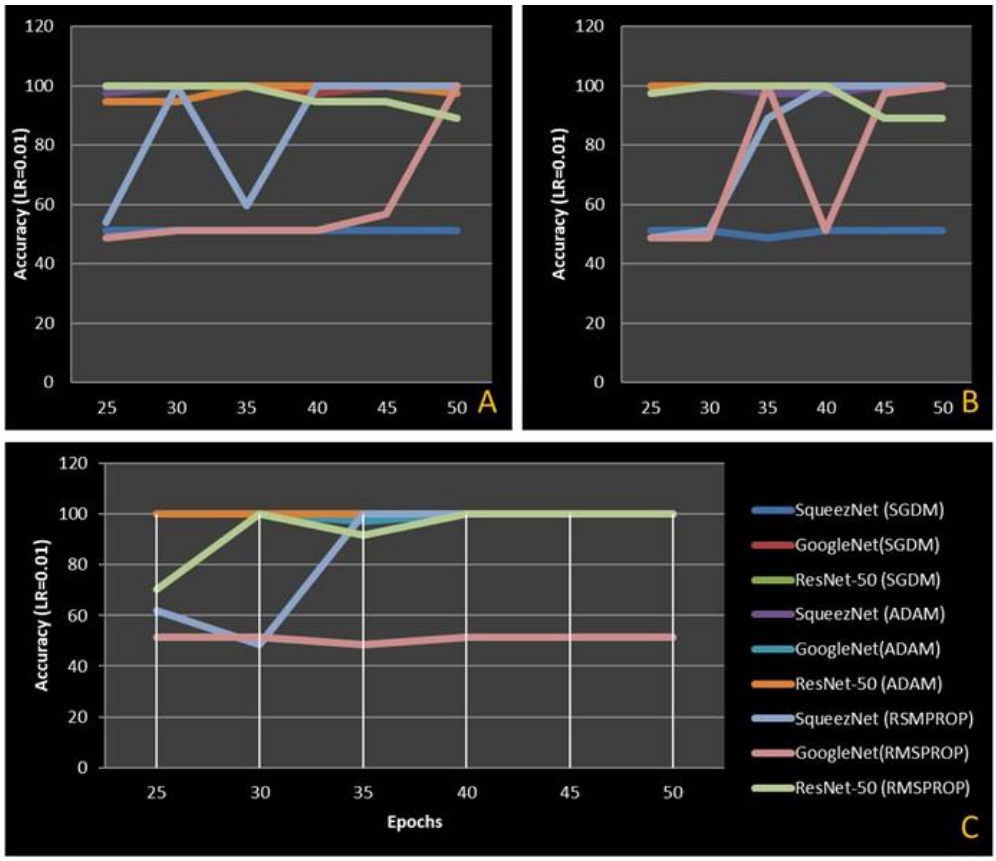

Figure 9. Performance analysis of DL vector distance (a) L2Norm (b) Global-L2Norm (c) Absolute-Value where LR = 0.01

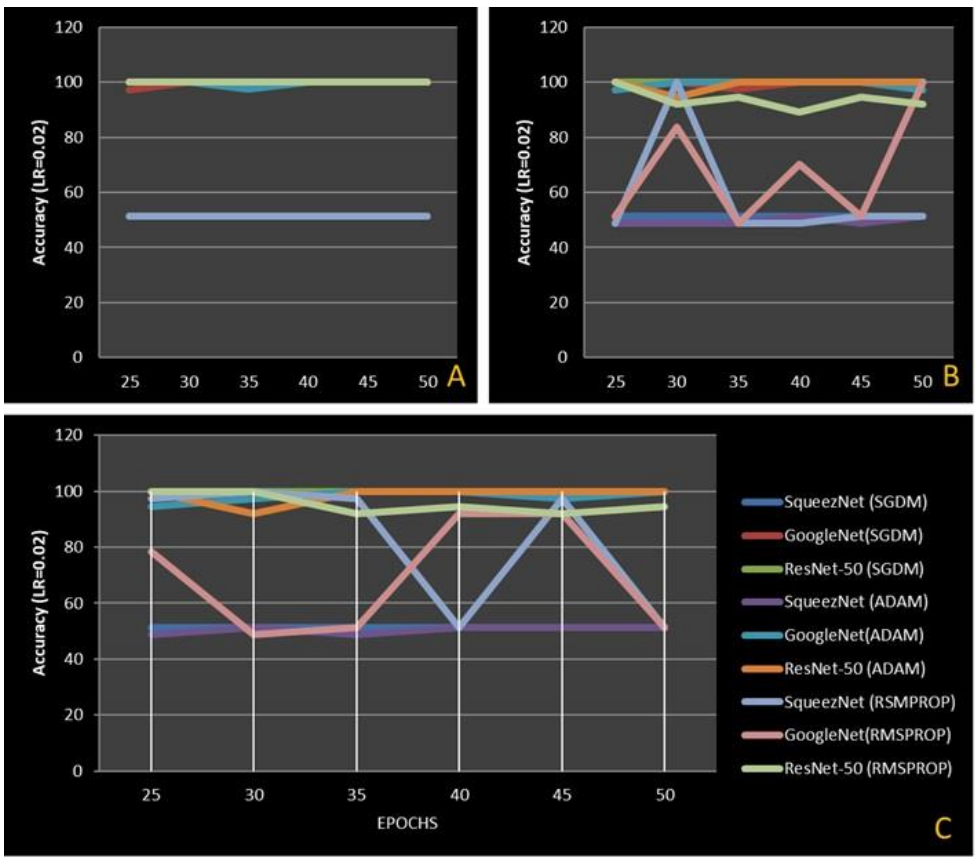

Figure 10. Performance analysis of DL vector distance (a) L2Norm (b) Global-L2Norm (c) Absolute-Value where LR = 0.02

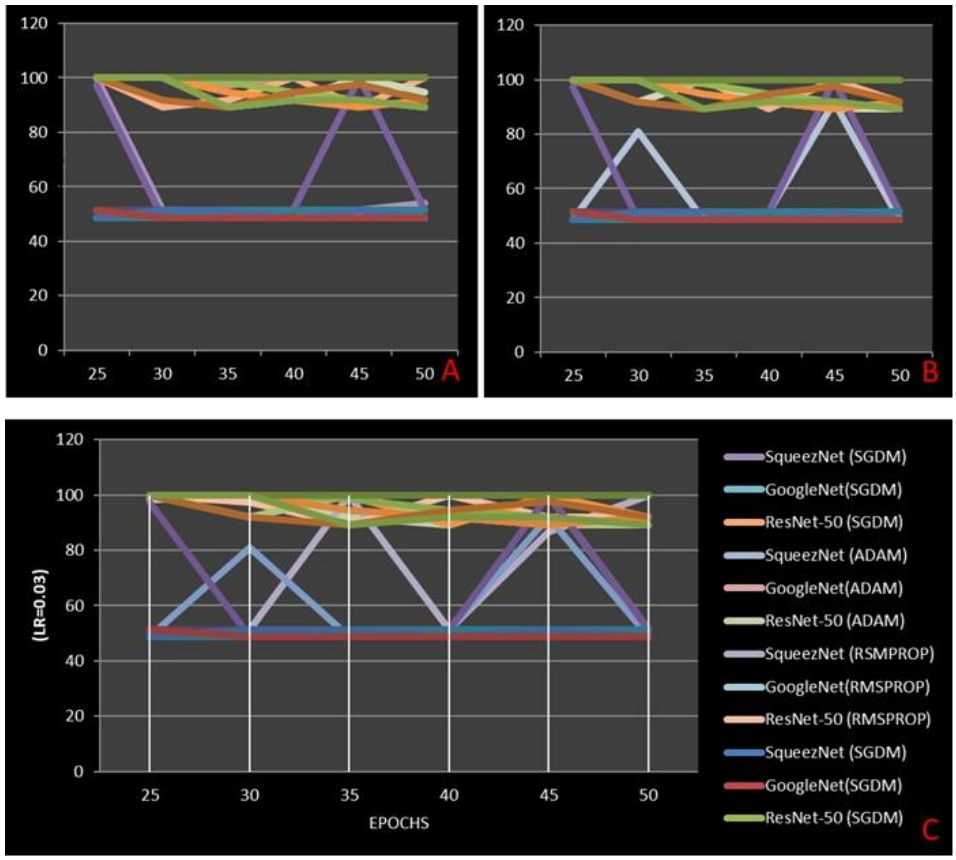

Figure 11. Performance analysis of DL network optimizers (a) L2Norm (b) Global-L2Norm (c) Absolute-Value where LR = 0.03

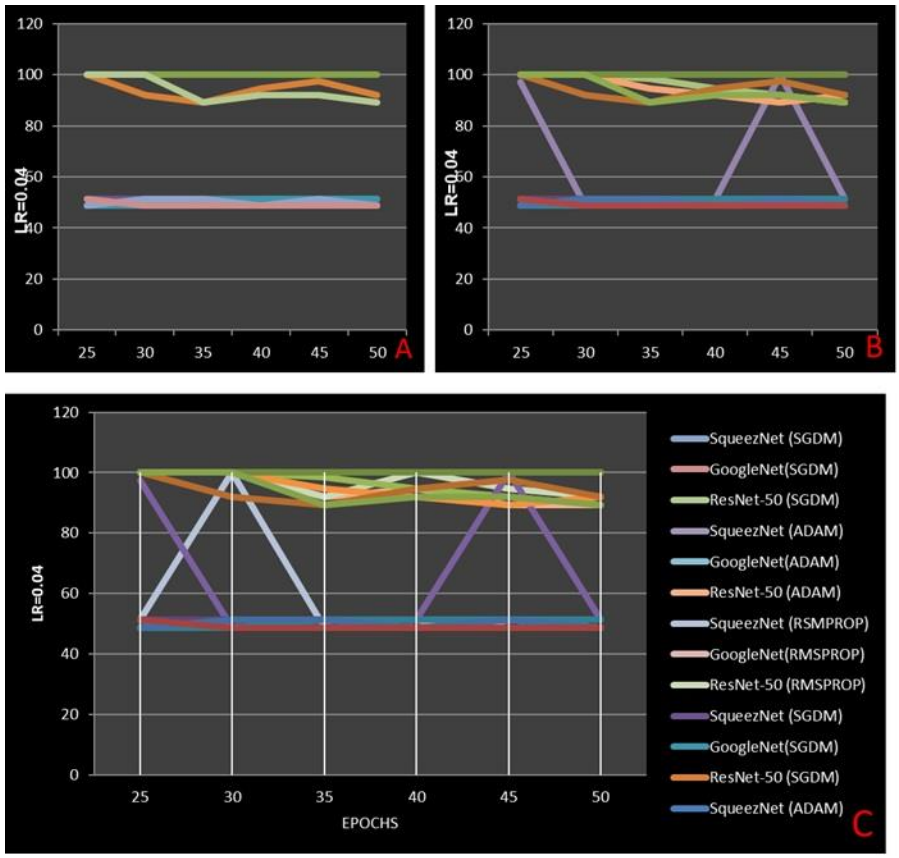

Figure 12. Performance analysis of DL network optimizers (a) L2Norm (b) Global-L2Norm (c) Absolute-Value where LR = 0.04

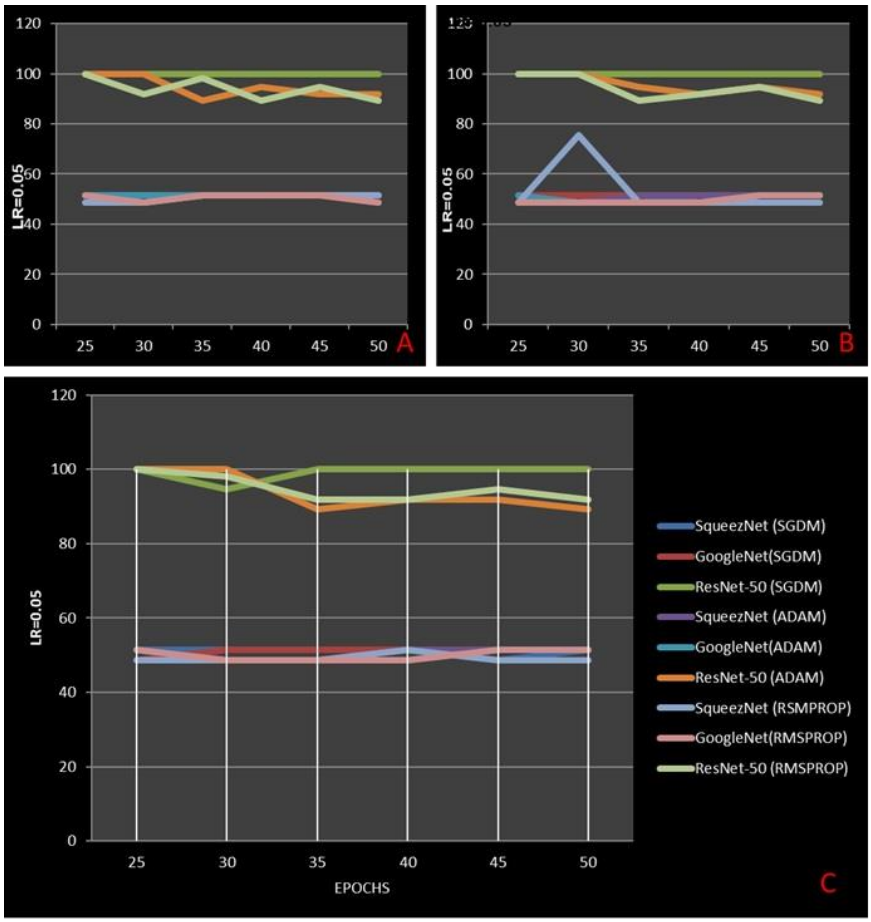

Figure 13. Performance analysis of DL network optimizers (a) L2Norm (b) Global-L2Norm (c) Absolute-Value where LR = 0.05

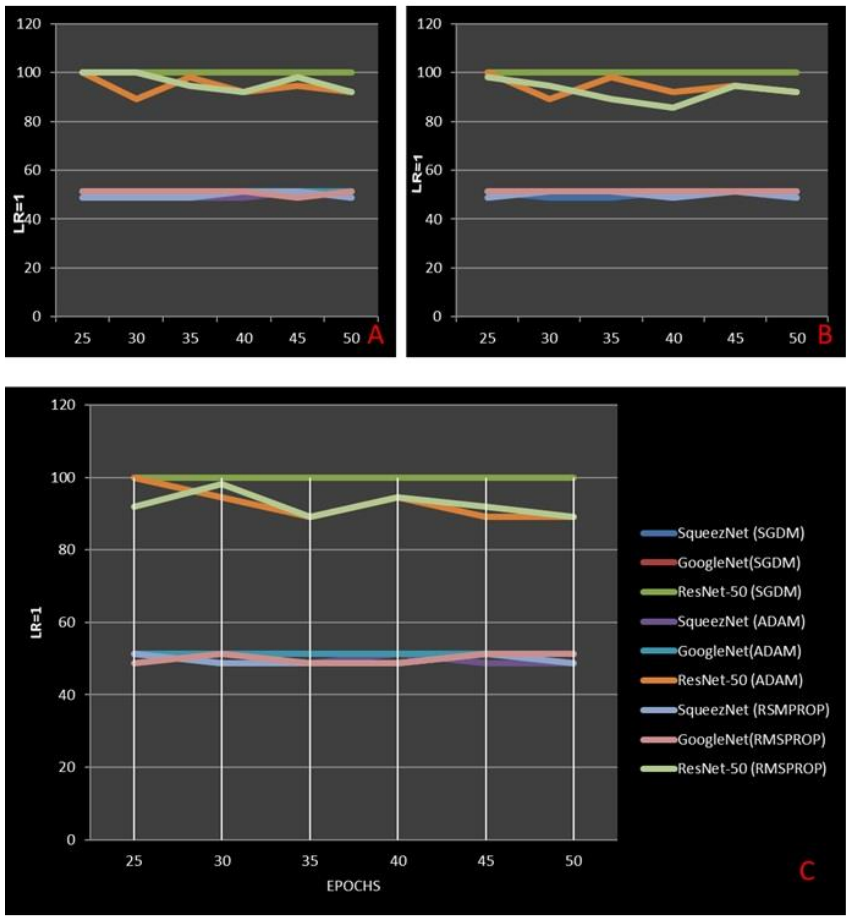

Figure 14. Performance analysis of DL network optimizers (a) L2Norm (b) Global-L2Norm (c) Absolute-Value where LR = 1

Adam is the suggested default algorithm and generally outperforms RMSProp and the mean epochs (25-50) and mean LR (0.01 to 1). Using the 'L2Norm' vector measurement method, ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) by 48.8% and GoogLeNet by 24.55%. In ADAM usage, ResNet-50 (ADAM) has 36.74% higher training accuracy than SqueezeNet and 28.1% higher than GoogLeNet. Finally, ResNet-50 (RMSProp) outperforms SqueezeNet (37.13%) and GoogLeNet (40.88%). ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) by 49.16.8% and GoogLeNet by 24.47% using the 'Global_L2Norm' vector measuring method. In ADAM usage, ResNet-50 (ADAM) has 38.29% higher training accuracy than SqueezeNet and 29.88% higher than GoogLeNet. The performance study of ResNet-50 (RMSProp) is 32.28% greater than SqueezeNet and 36.63% higher than GoogLeNet. ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) by 49.32% and GoogLeNet by 32.50% using the 'Absolute' vector measuring approach. ResNet-50 (ADAM) has 37.03% higher training accuracy than SqueezeNet (ADAM) and 29% higher than ResNet-50 (SGDM) outperforms SqueezeNet (SGDM) by 49.32% and 32.50% using the 'Absolute' vector measuring approach. Tables 3 and 4 show training classification accuracy statistics.

The objective is very straightforward in classifying three Curcuma longa disease classes. We have created a synthetic Curcuma longa dataset that exhibits 100% linear separability, distinguishing points inside a circle from those outside. The Curcuma longa dataset contains a significant amount of clean, well-prepared data, which is proportionate to the complexity of the model. The signal in the dataset is potent and evident. The model's architecture is appropriately designed and not overly intricate for the task. A basic logistic regression model or a modest neural network may be the appropriate instrument. The model achieved 100% training accuracy, attributed to its simplicity, rigorous data verification, and high validation performance. The disease detection problem in the Curcuma longa leaf dataset is linearly separable, and the clean dataset removes noise and leakage. The model maintains 99.8% accuracy on the hold-out validation set and 99.5% on the final test set. In addition, the test results were subjected to an ANOVA test and achieved a higher test result of 98.7%. The SqueezeNet with ADAM optimizer is perfectly suited for a very simple, clean, and linearly separable problem. The task was easy, and the model learned it flawlessly.

10. ANOVA TEST

An ANOVA test is conducted for the optimizers SGDM, ADAM, and RMSProp to determine the significant difference between the means of the SGDM, ADAM, and RMSProp values. For all the optimizers, the mean values are the same, which supports the null hypothesis; however, one of the mean values for SGDM, ADAM, or RMSProp may differ from the others. The above test is used to compare the accuracy, loss, and F1 score of the SGDM, ADAM, and RMSProp optimizers across the SqueezeNet, GoogLeNet, and ResNet-50 models to find the best accuracy among them. This analysis will offer information about which optimizer yields superior performance in terms of accuracy and efficiency. By evaluating the results obtained from each model, we can identify trends and make informed decisions on the best optimizer to use for detecting diseases in Curcuma longa leaves.

Figure 15 describes the ANOVA test results of SqueezeNet for the SGDM, ADAM, and RMSProp optimizers at 0 to 50 epochs each. It defines the difference in the optimizers' mean values for the true positive and false positive rates. While these results show performance variations among the optimizers, further analysis is necessary to determine the statistical significance of these differences. Additionally, exploring the impact of hyperparameter tuning on each optimizer could yield valuable insights into their effectiveness in different contexts. The accuracy of the test result SGDM is 99.44, ADAM is 97.00 and RMSProp is 99.16.

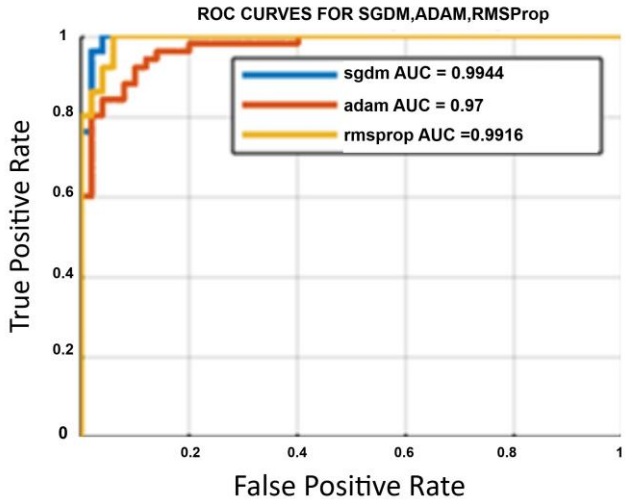

Figure 15. Anova test result of the SGDM, ADAM, RMSProp for the model SqueezeNet

Figure 16 describes the ANOVA test results of GoogLeNet for the SGDM, ADAM, and RMSProp optimizers at 0 to 50 epochs each. It defines the difference in the optimizers' mean values for the true positive and false positive rates. These findings indicate how each optimizer impacts model performance, particularly in terms of accuracy and reliability. The accuracy of the test result SGDM is 99.52, ADAM is 98.64 and RMSProp is 99.16.

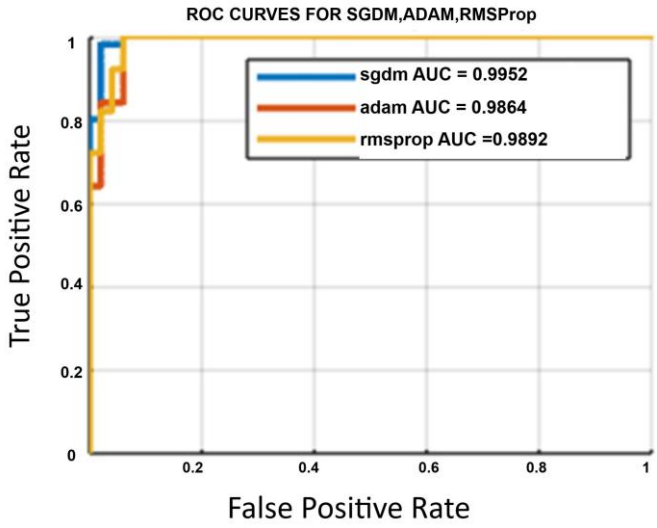

Figure 16. Anova test result of the SGDM, ADAM, RMSProp for the model GoogLeNet

Figure 17 describes the ANOVA test results of ResNet-50

for the SGDM, ADAM, and RMSProp optimizers at 0 to 50 epochs each. It defines the difference in the optimizers' mean values for the true positive and false positive rates. This analysis highlights how each optimizer impacts the model's performance, specifically in terms of accurately identifying true positives while minimizing false positives. By comparing the mean values, model can better yield the most effective results for ResNet-50 across the specified epochs. The accuracy of the test result SGDM is 99.52, ADAM is 98.04 and RMSProp is 99.20.

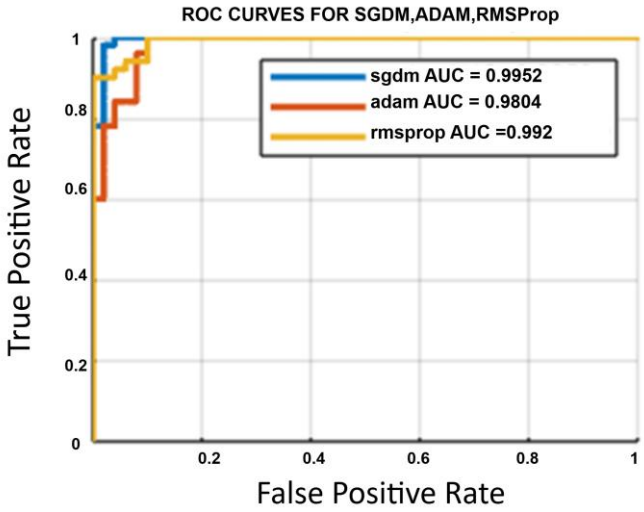

Figure 17. Anova test result of the SGDM, ADAM, RMSProp for the model ResNet-50

Figure 18 views the ADAM optimizer consistently showed lower performance across all evaluated architectures, resulting in the least test accuracy, indicating that SGD-based methods are more effective for these models over 0-50 epochs. In comparison, SGDM and RMSProp stood out as the leading methods, with SGDM demonstrating superior performance in SqueezeNet and GoogLeNet, whereas RMSProp had a slight edge in ResNet-50. SqueezeNet exhibited significant sensitivity to the choice of optimizer, revealing an accuracy gap exceeding 2% when compared to the top-performing optimizers. GoogLeNet consistently demonstrated strong performance across all optimizers while maintaining the same ranking. ResNet-50 demonstrated that both SGDM and RMSProp are highly effective for deeper networks, yielding nearly identical top outcomes. Although ANOVA indicates variations in mean values across the optimizers, additional analysis is required to determine statistical significance. The performance of ADAM could be notably enhanced through hyperparameter tuning, indicating that adjustments to parameters may influence the rankings.

The analysis of SqueezeNet, GoogLeNet and Resnet-50 using the ANOVA test revealed significant insights into the performance of various optimizers, including SGDM, ADAM, and RMSProp. By examining the true positive and false positive rates associated with each optimization method, it became evident that these variables are crucial in determining the efficacy of deep learning models SqueezeNet, GoogLeNet and Resnet-50. The nuanced differences observed highlight not only the strengths and weaknesses of each optimizer but also their impact on model accuracy and reliability. This comprehensive evaluation emphasizes the value of selecting appropriate optimization strategies to enhance model performance in Curcuma long a leaf disease detection. The accuracy of the test result SGDM is 99.48, Adam is 98.32 and RMSProp is 99.40.

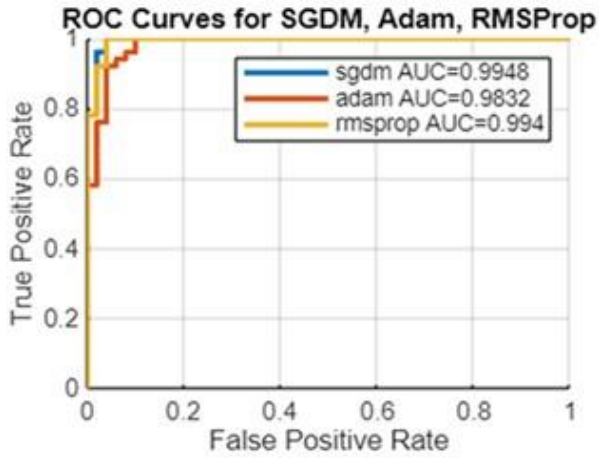

Figure 18. Anova test result of the SGDM, ADAM, RMSProp for the models SqueezeNet, GoogLeNet and resnet-50

11. DISCUSSION

The current study compares previous research, namely the work of Khan et al. [26], who conducted a comparative analysis of the performance of GoogLeNet, AlexNet, and SqueezeNet. By tweaking hyperparameters using electroencephalogram (EEG) data, they achieved accuracy rates of 94.99%, 94.61%, and 94.09%, respectively. The findings of the detection analysis indicate that AlexNet outperforms GoogLeNet and SqueezeNet. However, they concentrated solely on the epochs and learning rate, neglecting the importance of optimizers. The current study only fine-tunes the hyperparameters and also examines the influence of optimizers on training accuracy. Ullah et al. [27] specifically examined the performance of AlexNet, ResNet18, and SqueezeNet on a dataset consisting of 4333 photos belonging to eight distinct categories of road fractures. In this experiment, the training and testing images remained consistent throughout the epoch and iteration. The investigation's focus was not primarily on the choice of optimizer. The accuracy achieved with ResNet18 is just 85.2%. The proposed approach examined the GoogLeNet, SqueezeNet, and ResNet-50 models, with a primary focus on addressing training difficulties. We achieved this by fine-tuning hyperparameters, particularly emphasizing the usage of optimizers such as ADAM, RMSProp, and SGDM. Ashhar et al. [28] mostly looked into how well different deep learning models, such as GoogLeNet, SqueezeNet, DenseNet, ShuffleNet, and MobileNetV2, could classify lung tumours seen on a CT scan. They reached an accuracy of 94.53% using the GoogLeNet model. Their research did not take into account ResNet-50 and primarily concentrated solely on validation accuracy. Dahiya et al. [29] concentrated on training accuracy and used the Plant Village dataset, a dataset of 20,640 images representing 15 classes and 3 species: pepper, potato, and tomato. They applied this dataset to eight different deep learning architectures, namely AlexNet, GoogLeNet, MobileNet, ResNet 18, ResNet 50, ResNet 101, ShuffleNet, and SqueezeNet. Epochs, learning rate, mini batch

size, and optimizer were the hyperparameters used. The range of epochs utilized varies from 30 to 50, with only the ADAM and SGDM optimizers being employed, while RMSProp is excluded. Out of the eight deep learning architectures, GoogLeNet demonstrates superior performance in accurately identifying larger datasets. Only two optimizers and a maximum of three epochs constrain their work. The current work applied three types of optimizers, namely L2Norm, Global-L2Norm, and Absolute, on six epochs ranging from 25 to 50. We set the batch size to 32 and employed vector distance methods. However, Dahiya et al. said that GoogLeNet is the most effective classifier. In our research, we found that ResNet-50 is the optimal model for classifying the smaller sick leaf dataset, mostly because of the variation in the vector distance model. In their study, Wagle and Harikrishnan [30] utilized various deep learning models, including AlexNet, VGG16, GoogLeNet, MobileNetv2, and SqueezeNet, to identify tomato leaf illnesses. They found that the VGG16 model exhibited superior accuracy and precision for most of the tomato leaf classes. However, it is worth noting that the study had limited scope for hyperparameter tuning. We compare the performance analysis with numerous cutting-edge research works. Finally, the current study is highly informative since it incorporates all hyperparameters to evaluate the performance of common deep learning algorithms on a small, balanced dataset [31-34]. This research is distinctive due to its integration of deep learning architecture with optimizer and distance vector approaches (L2Norm, Global-L2Norm, and Absolute) while also adjusting epochs and learning rates. This approach resulted in improved classification performance on a smaller dataset while maintaining minimal time complexity. This original technical analysis demonstrates the uniqueness of the research. The current effort focuses primarily on medical plants rather than commercial crops and traditional object identification. Medical plants have become particularly important in the aftermath of the COVID-19 pandemic. The current investigation is limited to three specific pre-trained models for the sake of simplicity. However, it is possible to further train the current dataset using additional deep learning models such as EfficeintNet, AlexNet, VGG16, DarkNet, PANet, ShuffleNet, NasNet Xception, MobileNet-v2, and others. This will allow for an evaluation of the effectiveness of the presented results. The current study is beneficial for researchers with smaller and balanced datasets, as they can achieve higher accuracy by appropriately adjusting the hyper-parameters.

Pandey et al. [35] Identifying diseases in medicinal plants is essential for the quality and effectiveness of herbal therapeutics. Improvements in morphological observation, histological assessment, molecular biology procedures, and imaging modalities have enhanced illness identification. Contemporary technology, including high-throughput DNA sequencing and real-time PCR, improves speed, precision, and efficacy. This highlights the significance of continuous efforts for sustainable production of medicinal plants and accessibility of herbal medicine. Their study specifically focused on DNA sequencing, but the current study focused on training issues in deep learning techniques to achieve higher test accuracies.

Pushpa et al. [36] research investigates three hybrid deep-learning models for the real-time identification of medicinal plant species. The models employ VGG 16, MobileNet, MobileNetV2, and ResNet50 as feature extractors. The final feature vector is derived by consolidating the extracted features. The hybrid model 3 surpasses other models, exhibiting enhanced performance attributable to feature channel rescaling. The models are lightweight CNNs designed for mobile applications and are anticipated to be augmented to encompass rare medicinal plant species for enhanced identification and biodiversity conservation. But their research not focused on SqueezeNet and GoogLeNet and also not focused on distance vector methods, which is an important parameter to achieve higher test classification.

Sharma and Vardhan [37], study seeks to enhance the precision and efficacy of identifying medicinal plants utilized in traditional medicine. A substantial dataset of medicinal plants and leaves is utilized to develop a deep learning architecture known as the Attention-based Enhanced Local and Global Features Network (AELGNet). The architecture identifies salient aspects from the images, deriving fundamental characteristics for both local and global extraction. The research demonstrated that AELGNet surpasses 14 existing methodologies, serving as an effective instrument for the precise and rapid identification of medicinal plants and leaves in both medical and industrial contexts. This work not considered the traning issues, which is plays a significant role in medicinal plant leaf disease detection that addressed in the present work [38, 39].

Achieving elevated training and validation accuracy through the selection of suitable optimizers and acceptable vector algorithms for classifying Curcuma longa leaf disease detection datasets present a continual challenge for academics and practitioners. This study primarily emphasizes datasets of medicinal plant leaves. This study achieved an optimal accuracy of 100% utilizing the ADAM optimizer with L2Norm distance vector, specifically for balanced datasets of Curcuma longa leaves. For example, additional entities such as automobiles, structures, and individuals may not yield superior outcomes based on the current optimizer's recommendations. This study identified the optimal combinations by adjusting different optimizers and vector distances to get higher accuracies for Curcuma longa-like leaf diseases.

We have created a synthetic Curcuma longa dataset that exhibits 100% linear separability, in identifying and characterizing medicinal plants for novel pharmaceuticals and therapies. It facilitates individualized care, conservation efforts, traditional medicine, and addresses taxonomic deficiencies by identifying and monitoring endangered species while safeguarding traditional knowledge.

12. CONCLUSION

The impact of climate change on agriculture has been very severe on crop production for the past three decades. Especially the changes observed in the hydrological cycle have resulted in cloud bursts that led to heavy storms and floods in a short time. That implies the rapid spread of plant diseases and is reflected in lower crop yielding and resulted for farmers. In recent times the growth of medicinal plants has been in high demand due to the impact of COVID-19. Now-a-days the common prefer herbal medicines due to their lesser side-effects, as it is observed in the pandemic. Curcuma longa (Turmeric) is one of the important plant species that is widely used in traditional AYUSH and Allopathic treatments as well as for domestic purposes. Turmeric crop yield statistics for the

past decade reveal that crop production is deficient due to geo-environmental factors that cause diseases and pests to Turmeric crops. To improve the crop yield currently there is a need for a technical crunch that results in the early identification of diseases. The current work primarily focuses on a selection of the best DL network that suits perfectly to achieve cent percent accuracy for early disease detection in the Curcuma longa (Turmeric) 'Duggirala' variant image dataset. Each of the three DL networks was evaluated on the Turmeric dataset, revealing that the training accuracy of ResNet-50 is superior to that of GoogLeNet and SqueezeNet, respectively. Based on the analysis of the trial, it is strongly advised to use the ResNet-50 deep learning architecture for classifying diseases in medicinal plant images. The current suggestion is founded on the inclusion of hyperparameters such as Learning rate and Epochs, along with optimizers like ADAM, SGDM, and RMSProp, as well as vector distance algorithms such as L2Norm, Global-L2Norm, and Absolute. The current study exclusively employs three pre-trained models, although the dataset has the potential to be evaluated with alternative training models such as EfficeintNet, AlexNet, VGG16, DarkNet, PANet, ShuffleNet, NasNet Xception, MobileNet-v2 and others. This research helps researchers using smaller, balanced datasets to improve accuracy by modifying hyper-parameters.

REFERENCES

[1] Tsalis, T.A., Malamateniou, K.E., Koulouriotis, D., Nikolaou, I.E. (2020). New challenges for corporate sustainability reporting: United nations' 2030 agenda for sustainable development and the sustainable development goals. Corporate Social Responsibility and Environmental Management, 27(4): 1617-1629. https://doi.org/10.1002/csr.1910

[2] Khetrapal, S., Bhatia, R. (2020). Impact of COVID-19 pandemic on health system & sustainable development goal 3. Indian Journal of Medical Research, 151(5): 395-399. https://doi.org/10.4103/ijmr.IJMR_1920_20

[3] Martinez-Millana, A., Saez-Saez, A., Tornero-Costa, R., Azzopardi-Muscat, N., Traver, V., Novillo-Ortiz, D. (2022). Artificial intelligence and its impact on the domains of universal health coverage, health emergencies and health promotion: An overview of systematic reviews. International Journal of Medical Informatics, 166: 104855. https://doi.org/10.1016/j.ijmedinf.2022.104855

[4] Mattihalli, C., Gedefaye, E., Endalamaw, F., Necho, A. (2018). Plant leaf diseases detection and auto-medicine. Internet of Things, 1: 67-73. https://doi.org/10.1016/j.iot.2018.08.007

[5] Modak, M., Dixit, P., Londhe, J., Ghaskadbi, S., Devasagayam, T.P.A. (2007). Indian herbs and herbal drugs used for the treatment of diabetes. Journal of Clinical Biochemistry and Nutrition, 40(3): 163-173. https://doi.org/10.3164/jcbn.40.163

[6] Shankar, D., Patwardhan, B. (2017). Ayush for new India: Vision and strategy. Journal of Ayurveda and Integrative Medicine, 8(3): 137-139. https://doi.org/10.1016/j.jaim.2017.09.001

[7] Villena-Tejada, M., Vera-Ferchau, I., Cardona-Rivero, A., Zamalloa-Cornejo, R., Quispe-Florez, M., Frisancho-Triveño, Z., Abarca-Meléndez, R.C., Alvarez-Sucari, S.G., Mejia, C.R., Yañez, J.A. (2021). Use of medicinal plants for COVID-19 prevention and respiratory symptom treatment during the pandemic in Cusco, Peru: A cross-sectional survey. PLoS One, 16(9): e0257165. https://doi.org/10.1371/journal.pone.0257165

[8] Bhatt, V.D., Pandya, B.B., Joshi, C.G., Kunjadia, A.P. (2013). Curcuma longa: An alternative to antibiotics to combat mastits in cattle. Wayamba Journal of Animal Science, 578: 582-589.

[9] Vaughn, A.R., Branum, A., Sivamani, R.K. (2016). Effects of turmeric (Curcuma longa) on skin health: A systematic review of the clinical evidence. Phytotherapy Research, 30(8): 1243-1264. https://doi.org/10.1002/ptr.5640

[10] Razavi, B.M., Ghasemzadeh Rahbardar, M., Hosseinzadeh, H. (2021). A review of therapeutic potentials of turmeric (Curcuma longa) and its active constituent, curcumin, on inflammatory disorders, pain, and their related patents. Phytotherapy Research, 35(12): 6489-6513. https://doi.org/10.1002/ptr.7224

[11] Acharya N G Ranga Agricultural University Outlook Report. https://angrau.ac.in/downloads/AMIC/OutlookReports/2023_24/turmeric%20outlook%20-June-july-2023-24.pdf, accessed on Nov. 19, 2023.

[12] National turmeric board 2022-2023 production report. https://pib.gov.in/PressReleaseIframePage.aspx?PRID=1964083#:~:text=India%20is%20the%20largest%20producer,%25%20of%20global%20turmeric%20production., accessed on Nov. 19, 2023.

[13] Orchi, H., Sadik, M., Khaldoun, M. (2021). On using artificial intelligence and the internet of things for crop disease detection: A contemporary survey. Agriculture, 12(1): 9. https://doi.org/10.3390/agriculture12010009

[14] Thangaraj, R., Anandamurugan, S., Pandiyan, P., Kaliappan, V.K. (2022). Artificial intelligence in tomato leaf disease detection: A comprehensive review and discussion. Journal of Plant Diseases and Protection, 129(3): 469-488. https://doi.org/10.1007/s41348-021-00500-8

[15] Math, R.M., Dharwadkar, N.V. (2022). Early detection and identification of grape diseases using convolutional neural networks. Journal of Plant Diseases and Protection, 129(3): 521-532. https://doi.org/10.1007/s41348-022-00589-5

[16] Naeem, S., Ali, A., Chesneau, C., Tahir, M.H., Jamal, F., Sherwani, R.A.K., Ul Hassan, M. (2021). The classification of medicinal plant leaves based on multispectral and texture feature using machine learning approach. Agronomy, 11(2): 263.

[17] Arunaggiri Pandian, K., Sai Kumar, T.S., Thabasum Aara, S., Prabalakshmi, A. (2021). Identification of indian medicinal plants from leaves using transfer learning approach. In 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, pp. 980-987. https://doi.org/10.1109/ICOEI51242.2021.9452917

[18] Roopashree, S., Anitha, J. (2021). DeepHerb: A vision based system for medicinal plants using xception features. IEEE Access, 9: 135927-135941. https://doi.org/10.1109/ACCESS.2021.3116207

[19] Kuricheti, G., Supriya, P. (2019). Computer vision based turmeric leaf disease detection and classification: A step to smart agriculture. In 2019 3rd International

Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, pp. 545-549. https://doi.org/10.1109/ICOEI.2019.8862706

[20] Gogoi, A., Munda, S., Paw, M., Begum, T., Siddiqui, M. H., Gaafar, A.R.Z., Kesawat, M.S., Lal, M. (2023). Molecular genetic divergence analysis amongst high curcumin lines of golden crop (Curcuma longa L.) using SSR marker and use in trait-specific breeding. Scientific Reports, 13(1): 19690. https://doi.org/10.1038/s41598-023-46779-5

[21] Chen, W.L., Lin, Y.B., Lin, Y.W., Chen, R., Liao, J.K., Ng, F.L., Chan, Y.Y., Liu, Y.C., Wang, C.C., Chiu, C.H., Yen, T.H. (2019). AgriTalk: IoT for precision soil farming of turmeric cultivation. IEEE Internet of Things Journal, 6(3): 5209-5223. https://doi.org/10.1109/JIOT.2019.2899128

[22] Devisurya, V., Devi Priya, R., Anitha, N. (2022). Early detection of major diseases in turmeric plant using improved deep learning algorithm. Bulletin of The Polish Academy of Sciences. Technical Sciences, 70(2): e140689. https://doi.org/10.24425/bpasts.2022.140689

[23] Upendar, K., Agrawal, K.N., Chandel, N.S., Singh, K. (2021). Greenness identification using visible spectral colour indices for site specific weed management. Plant Physiology Reports, 26(1): 179-187. https://doi.org/10.1007/s40502-020-00562-0

[24] Wang, L., Wang, C., Sun, Z., Cheng, S., Guo, L. (2020). Class balanced loss for image classification. IEEE Access, 8: 81142-81153. https://doi.org/10.1109/ACCESS.2020.2991237

[25] Liu, W., Wu, Z., Wang, Y., Ding, H., Liu, F., Lin, J., Lin, G. (2024). LCReg: Long-tailed image classification with latent categories based recognition. Pattern Recognition, 145: 109971. https://doi.org/10.1016/j.patcog.2023.109971

[26] Khan, I.D., Khan, M.H., Farooq, O., Khan, Y.U. (2021). A comparative analysis of seizure detection via scalogram using GoogLeNet, AlexNet and SqueezeNet. In 2021 Smart Technologies, Communication and Robotics (STCR), Sathyamangalam, India, pp. 1-5. https://doi.org/10.1109/STCR51658.2021.9588862

[27] Ullah, A., Elahi, H., Sun, Z., Khatoon, A., Ahmad, I. (2022). Comparative analysis of AlexNet, ResNet18 and SqueezeNet with diverse modification and arduous implementation. Arabian Journal for Science and Engineering, 47(2): 2397-2417. https://doi.org/10.1007/s13369-021-06182-6

[28] Ashhar, S.M., Mokri, S.S., Abd Rahni, A.A., Huddin, A.B., Zulkarnain, N., Azmi, N.A., Mahaletchumy, T. (2021). Comparison of deep learning Convolutional Neural Network (CNN) architectures for CT lung cancer classification. International Journal of Advanced Technology and Engineering Exploration, 8(74): 126. https://doi.org/10.19101/IJATEE.2020.S1762126

[29] Dahiya, S., Gulati, T., Gupta, D. (2022). Performance analysis of deep learning architectures for plant leaves disease detection. Measurement: Sensors, 24: 100581. https://doi.org/10.1016/j.measen.2022.100581

[30] Wagle, S.A., Harikrishnan, R. (2021). A deep learning-based approach in classification and validation of tomato leaf disease. Traitement du Signal, 38(3): 699-709. https://doi.org/10.18280/ts.380317

[31] Rachmad, A., Setiawan, W., Rochman, E.M.S. (2023). Comparing the architecture of Convolutional Neural Network for corn leaves diseases image classification. In 1st International Conference on Neural Networks and Machine Learning 2022 (ICONNSMAL 2022), pp. 81-88. https://doi.org/10.2991/978-94-6463-174-6_9

[32] Aman, B.K., Kumar, V. (2023). Flower leaf image classification using deep learning techniques. In Recent Trends in Computational Intelligence and Its Application, pp. 51-60.

[33] Praveena, S., Pavithra, S.M., Kumar, A.D.V., Veeresha, P. (2024). CNN-based Indian medicinal leaf type identification and medical use recommendation. Neural Computing and Applications, 36(10): 5399-5412. https://doi.org/10.1007/s00521-023-09352-9.

[34] Uddin, A.H., Chen, Y.L., Borkatullah, B., Khatun, M.S., Ferdous, J., Mahmud, P., Yang, J., Ku, C.S., Por, L.Y. (2023). Deep-learning-based classification of Bangladeshi medicinal plants using neural ensemble models. Mathematics, 11(16): 3504. https://doi.org/10.3390/math11163504

[35] Pandey, B.N., Pandey, M.S., Pandey, B. (2024). An identification technique for diseases of medicinal plants. In 2024 7th International Conference on Contemporary Computing and Informatics (IC3I), Greater Noida, India, IEEE, pp. 535-540. https://doi.org/10.1109/IC3I61595.2024.10829079

[36] Pushpa, B.R., Jyothsna, S., Lasya, S. (2025). HybNet: A hybrid deep models for medicinal plant species identification. MethodsX, 14: 103126. https://doi.org/10.1016/j.mex.2024.103126

[37] Sharma, S., Vardhan, M. (2025). Aelgnet: Attention-based enhanced local and global features network for medicinal leaf and plant classification. Computers in Biology and Medicine, 184: 109447. https://doi.org/10.1016/j.compbiomed.2024.109447

[38] Banala, R.K., Duvvuru, R. (2025). A study on the difficulties of training with deep learning techniques to identify Curcuma longa leaf diseases. AIP Conference Proceedings, 3298(1): 020038.

[39] Banala, R., Duvvuru, R. (2025). Novel performance analysis of YOLOv5 and YOLOv8 for Curcuma longa leaf disease identification. Journal of Theoretical and Applied Information Technology, 103(5): 2071-2089. https://jatit.org/volumes/Vol103No5/30Vol103No5.pdf.