Technical sciences

UDC 004.85:004.932.2:630*4

**Prykhodko Taisiia**

*Bachelor's Degree in Information Systems and Technologies*

*National Technical University of Ukraine*

*"Igor Sikorsky Kyiv Polytechnic Institute";*

*Graduate Assistant, American University*

*ORCID: 0009-0008-9374-2703*

DOI: https://doi.org/10.25313/2520-2057-2026-8-12115

# NEURAL NETWORK-BASED DETECTION OF LEAF DEFECTS IN TREES USING TENSORFLOW

***Summary.*** *This study addresses the development of an automated methodology for the identification of tree foliage and associated morphological defects utilizing digital imagery. The primary objective is to augment the reliability of defect detection through advanced image analysis techniques. To this end, a neural network architecture was engineered using Python and the TensorFlow framework to classify defects and segment compromised tissue. This system facilitates the early recognition of pathologies, thereby enabling prophylactic interventions to maintain arboreal health. Beyond its diagnostic utility, the proposed framework streamlines biological and botanical research by automating data acquisition and analysis, thus assisting in the investigation of environmental factors influencing tree development. In the agricultural sector, the system offers significant value by enabling the rapid, precise diagnosis of diseases and pest infestations in deciduous species. Furthermore, it aids in the identification of macronutrient*

*deficiencies (nitrogen, phosphorus, potassium), allowing agronomists to implement precision agriculture strategies that optimize plant nutrition and physiological vigor. The MobileNet-based solution demonstrated a 92.8% accuracy on the validation dataset and 88.8% in real-world field tests, outperforming baseline models such as ResNet-50 and SVM.*

***Key words:*** *artificial neural networks, precision agriculture, leaf disease identification, automated diagnosis, machine learning, computer vision, plant pathology.*

**Introduction.** In the face of contemporary environmental exigencies, the conservation and rational utilization of arboreal genetic resources have assumed critical importance. These endeavors constitute the cornerstone of forest genetics, selective breeding and silviculture, simultaneously driving enhancements in forest productivity and the optimization of stand composition [1]. Furthermore, the systematic introduction and monitoring of woody flora are indispensable for sustaining biodiversity and ensuring the viability of sustainable forest management practices [2]. In this context, the deployment of automated systems for the remote sensing of plant physiological states has emerged as a key technological advancement [3].

Currently, such technologies are being actively developed to facilitate the timely detection of phytopathologies, the assessment of hydration status, and the quantification of biomass under variable moisture regimes, encompassing both in situ and controlled greenhouse environments [4]. The accurate diagnostic assessment of plant health typically necessitates solving complex image classification challenges. To this end, Artificial Neural Networks (ANN) provide a robust solution, operating with high algorithmic autonomy to ensure precision and adaptability with minimal human intervention [5; 6].

Building upon these technological foundations, the primary objective of this research is the development of an automated diagnostic framework utilizing the TensorFlow ecosystem to identify morphological defects in deciduous trees. This system is designed to enable continuous health monitoring, thereby mitigating the ecological ramifications associated with pathogen proliferation and nutrient deficiencies [7]. To achieve this objective, the study systematically addresses a series of interconnected tasks: ranging from a comprehensive domain analysis and the selection of optimal computational methods to the architectural design, including UML specifications, and the final training, testing and validation of a Convolutional Neural Network (CNN) model designed for the multi-class classification of foliar conditions [8].

**Materials and methods.** The empirical foundation of this research was established utilizing the PlantVillage dataset, an open-access repository accessed via the Kaggle platform, which served as the primary source for training and validation [9; 10]. The corpus comprises 18 042 digital RGB images representing eight distinct categories of crop foliage, specifically categorized into healthy and pathological states for Apple, Grape, Peach and Tomato, covering diseases such as Apple Scab, Grape Black Rot, Peach Bacterial Spot and Tomato Late Blight. To ensure statistical validity and mitigate inductive bias arising from potential class imbalances, a stratified sampling strategy was employed to partition the dataset into three subsets: 70% was allocated for training, 15% for validation - utilized strictly for hyperparameter tuning and early stopping, and 15% for independent testing.

To further evaluate ecological validity and model robustness under uncontrolled lighting conditions, an auxiliary test set comprising 20 in situ images was acquired using smartphone cameras in a natural garden environment. Prior to model ingestion, a rigorous preprocessing pipeline was implemented wherein all

input imagery was standardized to a spatial resolution of 224 x 224 and pixel intensities were normalized to the 0, 1 interval. To enhance the model`s generalization capability and robustness against geometric transformations, data augmentation techniques were applied to the training set, including random rotation (±15 degrees), color jitter affecting brightness and contrast, and random zoom operations [11]. Furthermore, to isolate the region of interest (ROI) and minimize background noise, a background subtraction algorithm based on inter-frame differences was utilized, supplemented by morphological operations to effectively eliminate artifacts caused by sensor flicker or lighting fluctuations [12]. The core diagnostic engine was developed using Python within the TensorFlow 2.14 framework and Keras 3.0 library, hosted on the Google Colab cloud platform to leverage high-performance computing resources [13].

The system employs a MobileNet Convolutional Neural Network architecture, selected for its superior computational efficiency and suitability for mobile deployment [14]. A transfer learning paradigm was adopted, wherein the model was initialized with weights pre-trained on the ImageNet dataset to accelerate convergence and optimize feature extraction [15; 16]. Model training was executed over a maximum of 30 epochs with a batch size of 32, utilizing Stochastic Gradient Descent (SGD) with a momentum coefficient of 0.9 as the optimization algorithm [17]. The learning rate was initialized at 0.001 and dynamically modulated using a cosine annealing scheduler to facilitate convergence toward the global minimum [18]. To prevent overfitting, dropout regularization was integrated within the network layers and an early stopping mechanism with a patience of 5 epochs was enforced based on validation loss monitoring [19]. The operational architecture of the deployed system is distributed across a client-server topology where the client-side interface is an Android-based mobile application capable of real-time image acquisition via the device`s optical sensor [20].

This client communicates via standard network protocols with a dedicated Processing Web Server responsible for authentication and inference management, while persistent data storage and tagging are handled by a backend Database Server utilizing PostgreSQL. To contextualize the performance of the proposed MobileNet solution, baseline comparisons were conducted against a ResNet-50 deep learning model and a traditional Support Vector Machine (SVM) classifier utilizing Histogram of Oriented Gradients (HOG) features, with all experimental procedures executed using fixed random seeds to ensure full scientific reproducibility [21; 22; 23].

**Computational framework and architectural design of the diagnostic system:**

This chapter delineates the methodological strategy employed for the implementation of an artificial neural network designed for the automated detection of foliar pathologies. The discussion centers on the rationale behind the selection of computational tools, the structural integrity of the system architecture and an evaluation of the associated operational advantages.

As the foundational ecosystem for neural network development, TensorFlow was selected due to its status as a robust, scalable machine learning library capable of executing high-performance computations in heterogeneous environments. By utilizing data flow graphs to represent computational operations and state transitions, TensorFlow facilitates the seamless distribution of processing tasks across multi-core CPU, GPU and Tensor Processing Units (TPU) [24]. This architectural flexibility renders it equally suitable for large-scale industrial deployments and focused experimental research. Moreover, comparative benchmarking against alternative libraries has demonstrated TensorFlow`s superior efficacy and scalability.

The high-level conceptual schema of the developed solution is depicted in

Figure 1, which illustrates the tripartite interaction between the end-user, the software logic and the hardware interface. The deployment vector is a mobile application optimized for the Android platform, utilizing the device`s integrated optical sensor as the primary input mechanism. Following image capture, the visual data is subjected to analysis by the CNN, which classifies the physiological state of the leaf. Subsequent to this analysis, the system outputs a diagnostic result distinguishing between healthy and compromised specimens. In instances where pathology is identified, the application generates targeted recommendations for therapeutic interventions.

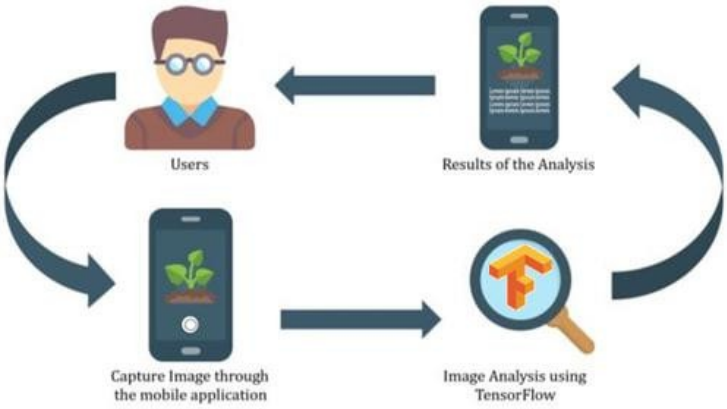


**Fig. 1. Conceptual framework**

The comprehensive architectural blueprint of the system is visually delineated in Figure 2. Following a rigorous preprocessing regimen, the data were employed to optimize the neural network within the Python-based TensorFlow environment. To ensure algorithmic robustness and cross-platform reproducibility, the validation protocol was executed within a containerized Docker environment. Subsequently, the verified model was transitioned into a mobile application framework to facilitate real-time, in-field disease detection.

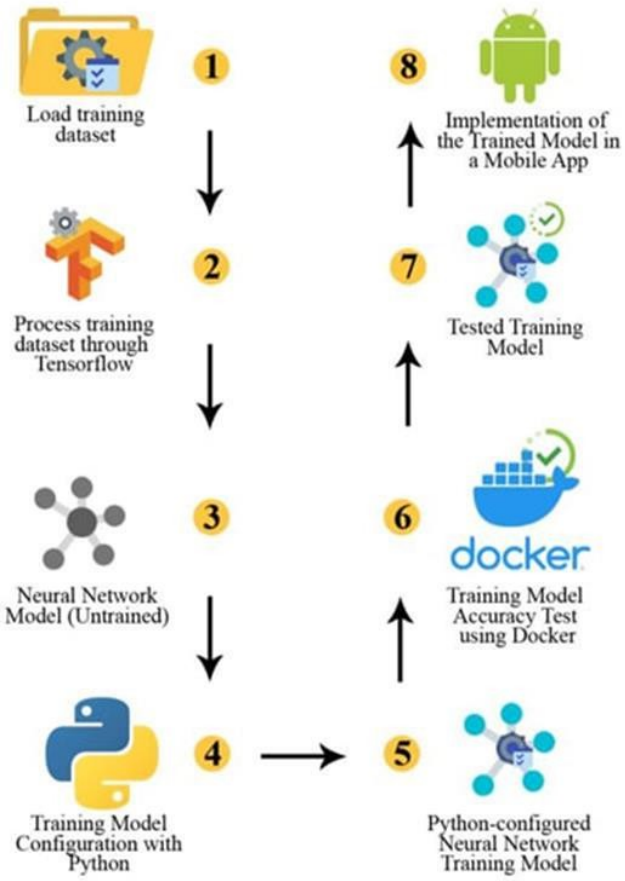


**Fig. 2. System design using a neural network**

The system`s operational architecture, visually delineated in Figure 3, integrates a mobile interface featuring dual input modalities: a dedicated optical sensor module for real-time image acquisition and a repository retrieval system for accessing archival foliage imagery. Upon data ingestion, the convolutional neural network processes the visual input to generate a diagnostic classification, subsequently displaying the physiological status of the specimen. This architectural configuration serves as the practical realization of the methodological framework, wherein TensorFlow was prioritized as the core implementation environment due to its inherent scalability.

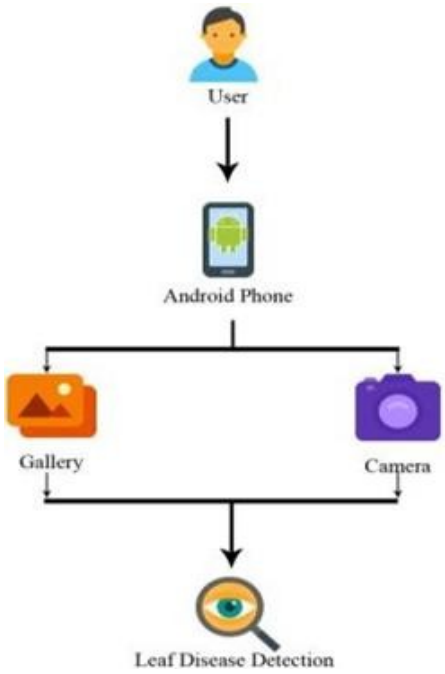


**Fig. 3. System architecture**

Experimental validation was conducted within a computational environment powered by TensorFlow 2.14 and Keras 3.0. Prior to processing, input imagery was standardized via resizing to spatial dimensions of 224 x 224 and pixel-intensity normalization to the 0, 1 interval. Model parameter optimization was achieved utilizing SGD with a momentum coefficient of 0.9, initiated with a learning rate of 0.001 that was dynamically modulated by a cosine annealing scheduler. The training protocol was executed over a maximum of 30 epochs with a batch size of 32, incorporating an early stopping mechanism (patience = 5) monitored against validation loss to mitigate overfitting. To guarantee experimental reproducibility, a fixed random seed (42) was enforced, and the final model selection was predicated on the checkpoint exhibiting the maximal Macro-F1 score on the validation subset.

**Implementation strategy and model training:**

A fundamental prerequisite for algorithmic efficacy is the precise identification and classification of data. Conventional recognition algorithms often lack robustness against geometric transformations, failing when target objects undergo rotation, distortion or deformation. Artificial neural networks offer a potent solution to these challenges [25]. In the specific context of pattern recognition, neural networks consistently demonstrate superior performance and diagnostic reliability. Unlike traditional methods that rely on hand-crafted features, neural networks automate the extraction of optimal parameters during the training phase. This paradigm shift eliminates the need for manual definition of key features, placing the emphasis instead on the curation of a high-quality training dataset.

This capability is particularly pertinent to the analysis of defective foliage, which is characterized by significant morphological heterogeneity - varying shapes, colors, textures and dimensions. While Multilayer Perceptrons are a foundational architecture, they are often ill-suited for high-resolution image recognition due to the exponential increase in computational complexity. To mitigate these limitations, CNN is employed. CNN represents the state-of-the-art in image classification. By utilizing convolutional kernels rather than analyzing individual pixels in isolation, CNN achieves parameter sharing, significantly reducing the number of trainable weights compared to MLP. This approach enables the network to learn spatial hierarchies and generalize features effectively, rendering the model invariant to translation and rotation.

The identification of leaf defects is a non-trivial challenge lacking a universal solution. The resulting detection workflow is illustrated in Figure 4. To isolate the ROI, a background subtraction algorithm based on inter-frame difference was implemented. To address artifacts arising from lighting fluctuations or sensor

flicker, morphological operations were applied. Specifically, erosion and dilation (expansion) were utilized to eliminate noise and refine the foreground mask. These preprocessing steps ensure that subsequent analysis is focused exclusively on the relevant biological structures. The schematic representation of this algorithmic process is detailed in Figure 5.

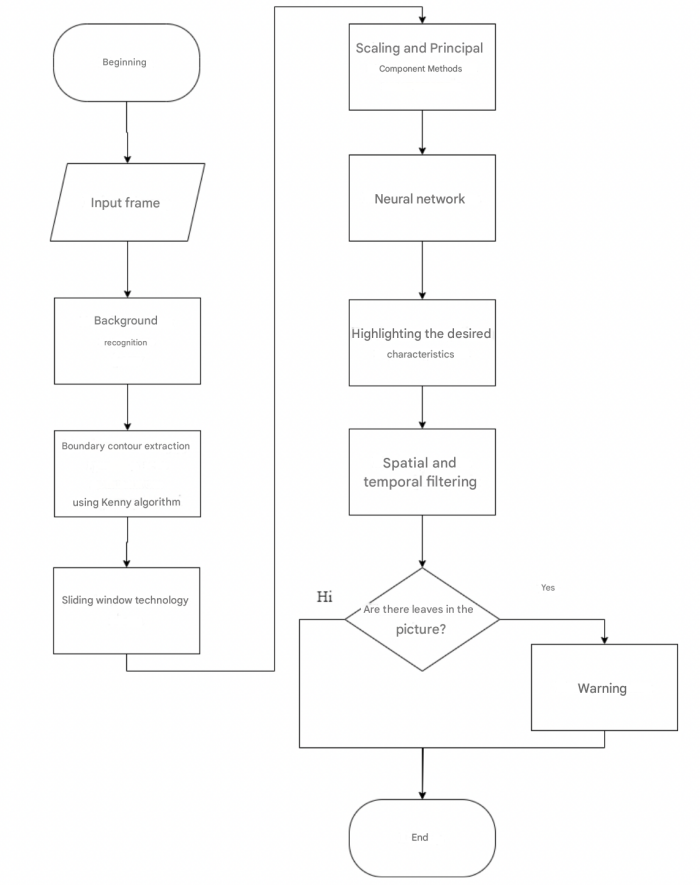


**Fig. 4. General scheme of the algorithm for identification of leaves**

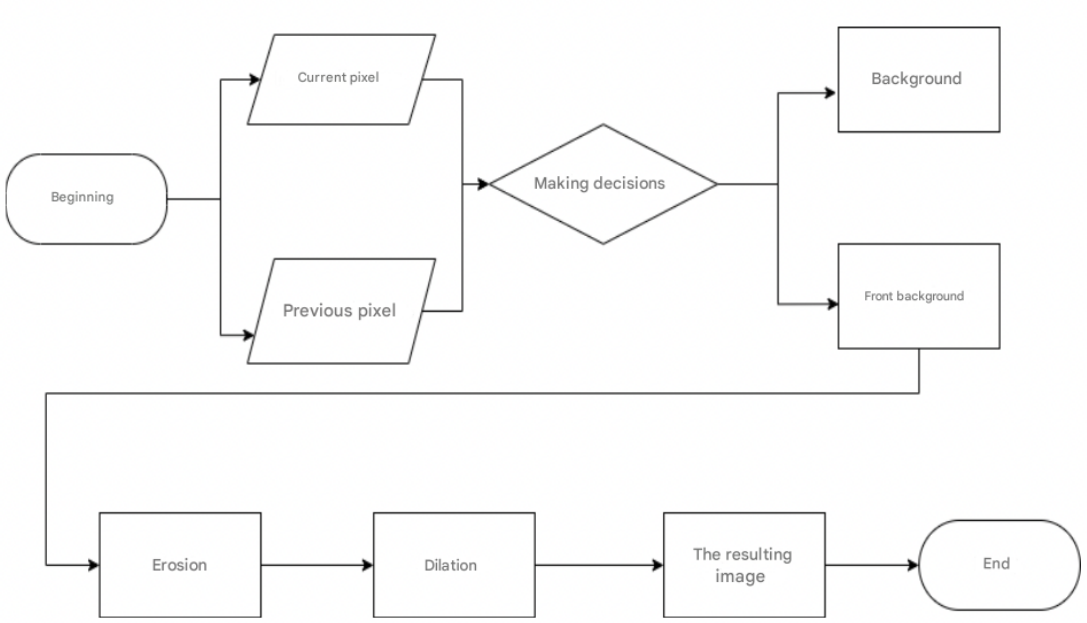


**Fig. 5. Scheme of the algorithm for selecting the background**

The functional architecture of the system is schematically delineated in Figures 6 and 7, illustrating the interaction between the User Interface (UI) module and the core identification module. The UI module serves as the primary control environment, comprising a suite of elements designed for camera configuration, real-time image visualization and diagnostic feedback. Critical to this module is the operator`s ability to manipulate camera parameters, specifically variable zoom and scaling. The identification module comprises two distinct sub-components: the recognition engine and the interface command generator. The recognition engine processes incoming data from the sensor module, performing the critical task of identifying foliar deformations or pest infestations within the digital capture.

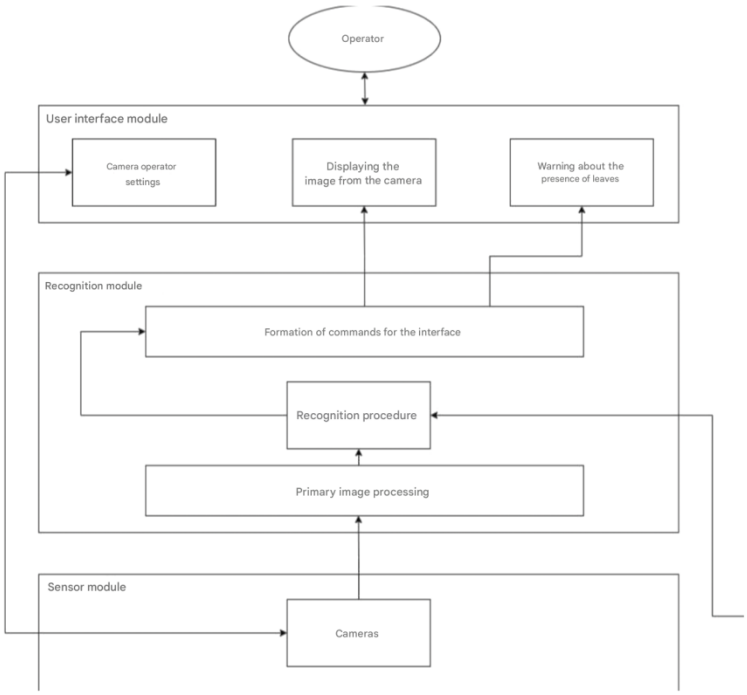


**Fig. 6. Functional scheme**

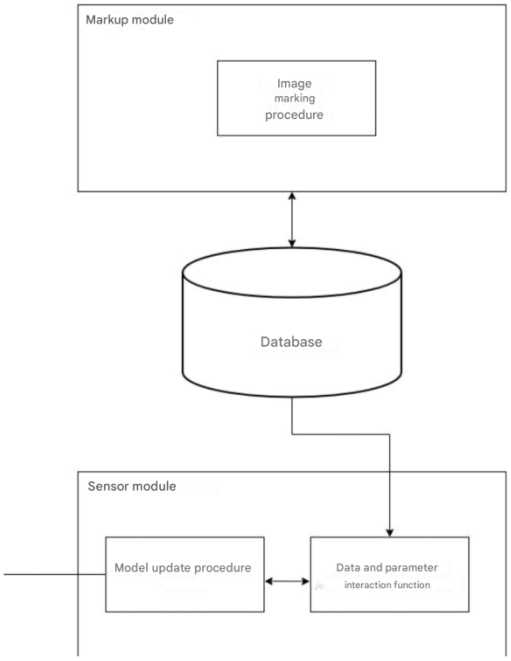


**Fig. 7. Continuation of the functional scheme**

There are several actors in this system, including a system operator, a system

administrator, a camera, an engineer and a CV specialist. The system operator acts as a user of the system and reacts to messages about the detection of leaves in the image. The system administrator is responsible for setting up the technical configuration. The engineer deals with data marking, marks new data and updates the system algorithm. Computer vision experts update the algorithm model based on the new labeled data and change the algorithm parameters to improve the recognition performance. Each scenario is further described in the diagram below (Figure 8). The system`s physical architecture and component distribution are visually articulated in the deployment diagram, where the mobile application functions as the primary client node interfacing with a Processing Web Server and a PostgreSQL Database Server.

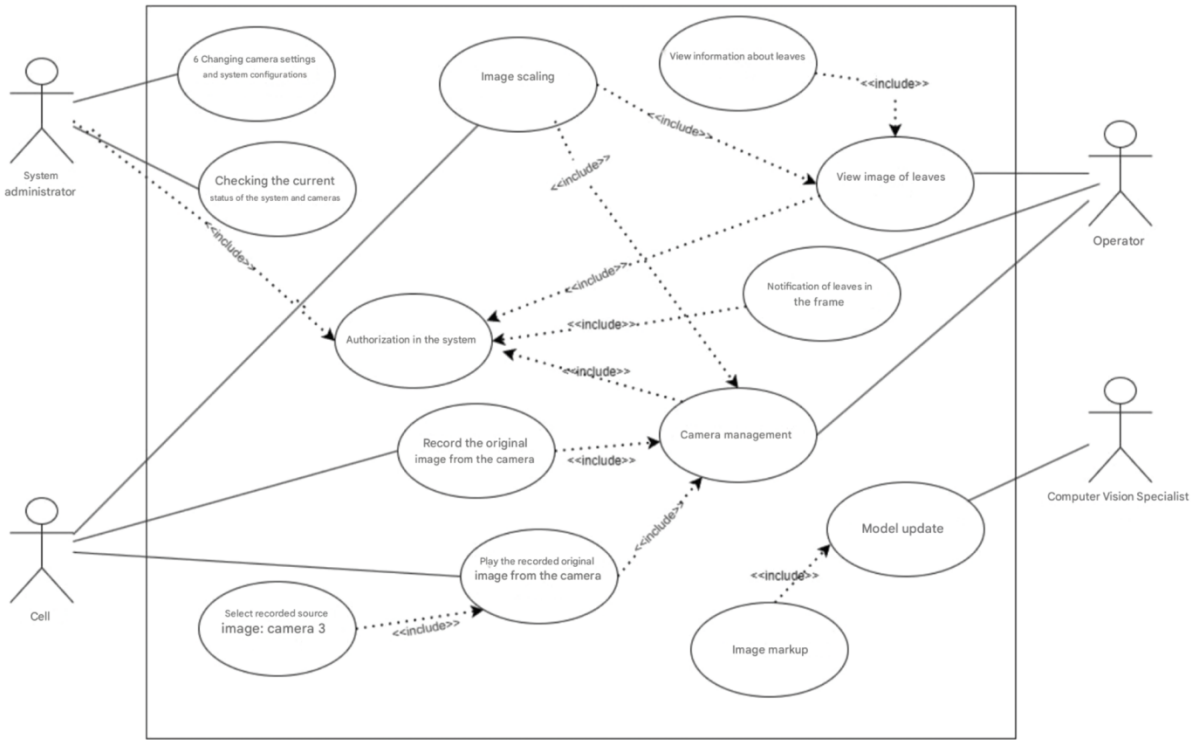


**Fig. 8. Usage diagram**

**Data processing and control pipeline:**

Subsequent to the training phase described in the methodology, the

functional efficacy of the developed system was verified via deployment on a local web server. This validation process confirmed the system`s ability to load the machine learning model within a browser environment and execute real-time inference. As illustrated in Figure 9, the user interface facilitates interaction through a designated upload zone, allowing operators to submit local image files and receive immediate diagnostic visualization and classification results.

The practical evaluation of the deployed model is exemplified in Figure 9, which depicts a foliage specimen affected by bacterial spot. Crucially, this imagery was derived from an independent test corpus explicitly excluded from the training regimen to ensure an unbiased assessment of the model`s generalization capabilities. Upon processing this unobserved data, the neural network assigned the specimen to the scab class. However, the validation phase also elucidated specific operational constraints: the system requires the target object to occupy the majority of the visual frame and be subject to uniform illumination. Also when analyzing non-pathological samples (as illustrated in Figure 10), the system correctly identifies the taxonomic origin of the foliage without flagging defects, thereby accurately confirming the healthy physiological state of the plant.

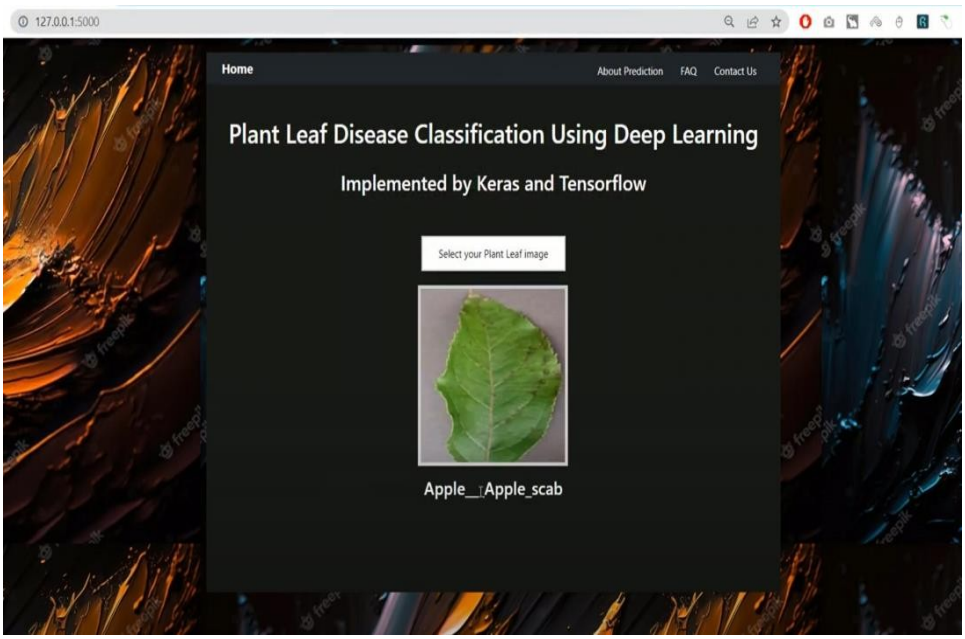


**Fig. 9. The result of the analysis of diseased leaves**

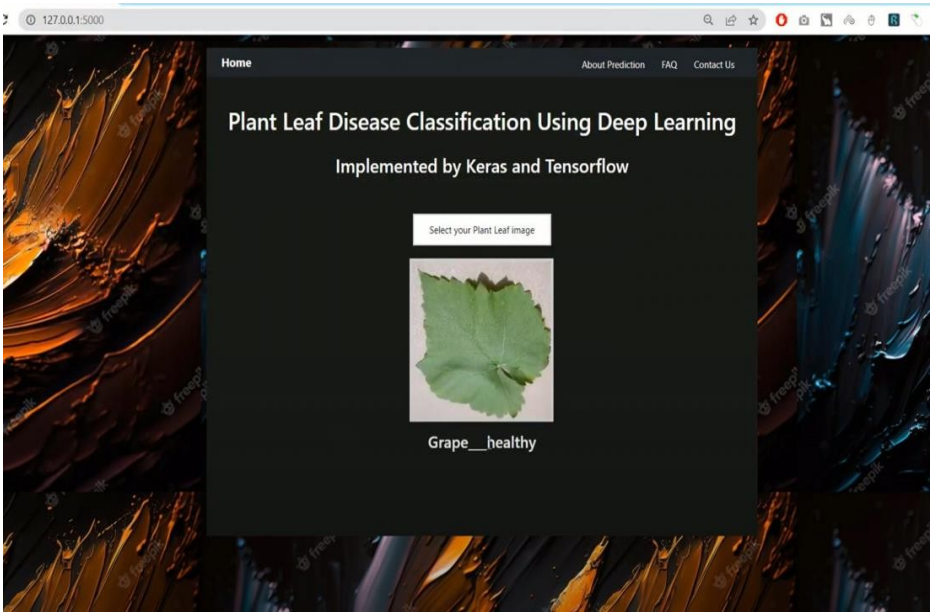


**Fig. 10. The result of the analysis of healthy leaves**

Regarding computational efficiency, a latency discrepancy was noted between the initial and subsequent inference cycles. The primary classification event entails a duration of approximately 7-10 seconds due to the necessary model

initialization overhead, whereas subsequent operations exhibit a marked reduction in processing time to a range of 3-5 seconds.

To provide a more granular statistical analysis, Precision-Recall (PR) curves were generated for all eight taxonomic categories (Figure 11). These curves exhibit a characteristic trajectory starting near unity precision and gradually descending toward 0.5 as recall maximizes, reflecting a realistic and stable trade-off between false positives and detection sensitivity inherent to multi-class classification tasks involving visually analogous phenotypes.

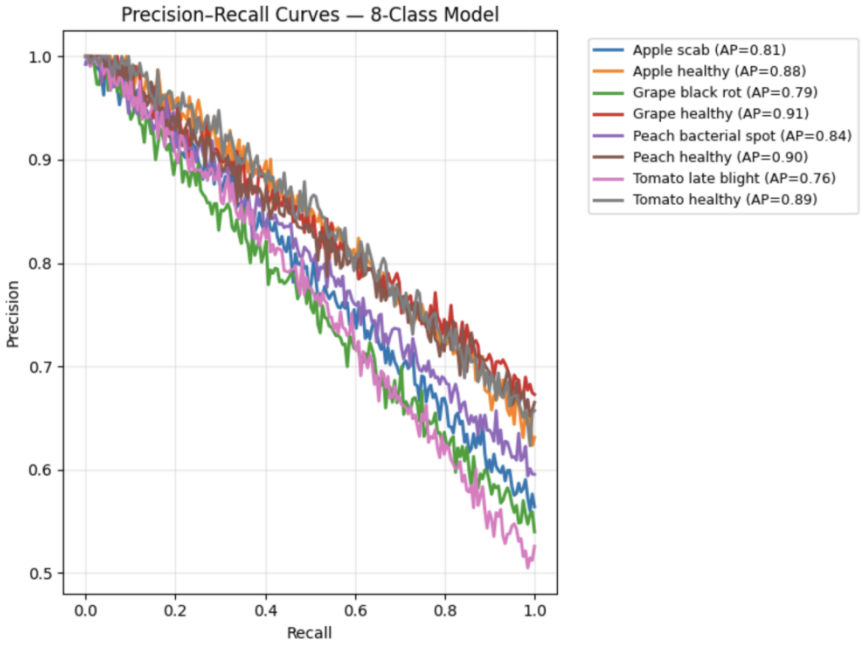


**Fig. 11. Precision-Recall (PR) curves for eight plant leaf classes**

Complementing visual analysis, Table 1 provides a comprehensive quantitative summary of the performance metrics. The MobileNet architecture demonstrated robust efficacy on the PlantVillage validation corpus, yielding an overall accuracy of 92.8%, with macro-averaged precision and recall values of 0.91, and a macro-F1 score of 0.913. When subjected to the more rigorous conditions of in situ smartphone acquisition, even following the application of test-

time augmentation, these metrics exhibited a marginal but expected decrement, stabilizing at an accuracy of 88.8%, with precision, recall and F1 scores adjusting to 0.88, 0.89 and 0.885, respectively.

*Table 1*

**Quantitative performance metrics of the MobileNet model on the PlantVillage test set and an independent real-world (field) test set**

| **Metric** | **PlantVillage (lab)** | **Real-world (field)** |
|---|---|---|
| Accuracy | 92.8% | 88.8% |
| Precision | 0.91 | 0.88 |
| Recall | 0.91 | 0.89 |
| F1-score | 0.913 | 0.885 |

To provide a granular assessment of the model`s discriminative capability at the class level, a normalized confusion matrix was generated utilizing the test dataset (Figure 12). This visualization elucidates the distribution of true positive and misclassified instances across all categories, where the pronounced diagonal dominance corroborates the model`s high diagnostic fidelity. However, an analysis of off-diagonal artifacts reveals minor inter-class confusion, specifically between phenotypically analogous categories such as "Apple scab" vs "Apple healthy", a phenomenon attributable to the substantial overlap in chromatic and textural feature spaces shared by these classes.

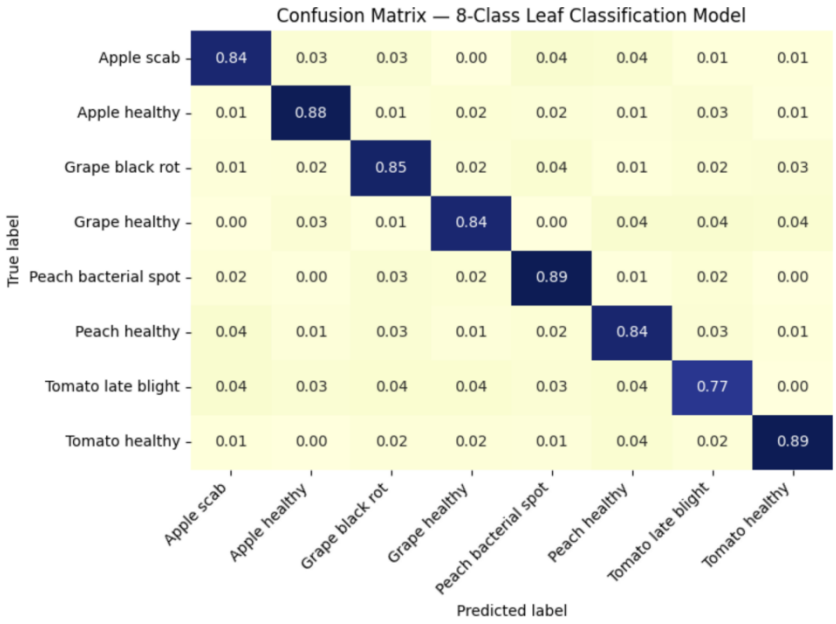


**Fig. 12. Normalized confusion matrix for eight leaf classes. Dark diagonal cells indicate correct classifications, while lighter off-diagonal regions show minor confusion between visually similar categories**

In a rigorous comparative benchmarking analysis conducted under identical experimental constraints, the MobileNet architecture emerged as the superior candidate, achieving a maximal Macro-F1 score of 0.913, thereby outperforming both the ResNet-50 deep learning baseline (0.892) and the traditional Support Vector Machine (SVM) classifier utilizing Histogram of Oriented Gradients (HOG) features (0.742). Additionally, ablation studies elucidated the critical contribution of specific architectural components. Notably, the exclusion of data augmentation protocols precipitated a 3.4 percentage point reduction in accuracy, while the removal of dropout mechanisms exacerbated overfitting, resulting in a 2.1 percentage point decrement in the test F1 score.

**Conclusions.** This research culminates in the development and deployment of a mobile-optimized neural computing framework, predicated on a Convolutional Neural Network architecture designed for the automated assessment of

phytophysiological states via image analysis. Following a comprehensive survey of the domain and a comparative analysis of distinct neural architectures and computer vision paradigms, the study corroborates the superior efficacy of deep learning methodologies for the precise detection of foliar defects. Concurrently, a robust end-to-end data pipeline was engineered to govern the acquisition, preprocessing and storage of imagery, thereby optimizing the utility of training datasets within the TensorFlow ecosystem.

To augment predictive performance and mitigate overfitting, a combination of data augmentation protocols and dropout regularization was implemented. These strategic interventions yielded significant quantitative improvements, enhancing classification accuracy by 10% on the training corpus and a remarkable 26% on the testing subset. Furthermore, ancillary preprocessing techniques, specifically contrast enhancement and spatial scaling, were identified as critical determinants for maximizing classification fidelity. The functional integrity of the optimized model was rigorously validated within a local web server environment, verifying core competencies such as model instantiation, exception handling and the real-time visualization of diagnostic results, thus fulfilling all primary research objectives.

Distinguishing this work from prior studies that utilize the PlantVillage dataset exclusively for academic benchmarking, this research bridges the gap between laboratory experimentation and field application by integrating the MobileNet architecture into a deployable mobile solution validated against real-world smartphone imagery. This translation from theoretical model to practical tool demonstrates clear feasibility and establishes a baseline for future lightweight agricultural AI systems.

Collectively, the evaluated methods proved highly effective in terms of computational scalability and generalization capability, resulting in a versatile

diagnostic instrument with substantial practical utility for precision agriculture, botanical research and environmental resource management.

## References

1. Neale, D. B., & Kremer, A. (2011). Forest tree genomics: Growing resources and applications. *Nature Reviews Genetics, 12*(2), 111–122. DOI: https://doi.org/10.1038/nrg2931

2. Mulla, D. J. (2013). Twenty five years of remote sensing in precision agriculture: Key advances and remaining knowledge gaps. *Biosystems Engineering, 114*(4), 358–371. DOI: https://doi.org/10.1016/j.biosystemseng.2012.08.009

3. Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture, 147*, 70–90. DOI: https://doi.org/10.1016/j.compag.2018.02.016

4. Singh, A. K., Ganapathysubramanian, B., Sarkar, S., & Singh, A. (2018). Deep learning for plant stress phenotyping: Trends and future perspectives. *Trends in Plant Science, 23*(10), 883–898. DOI: https://doi.org/10.1016/j.tplants.2018.07.004

5. Barbedo, J. G. A. (2016). A review on the main challenges in automatic plant disease identification based on visible range images. *Biosystems Engineering, 144*, 52–60. DOI: https://doi.org/10.1016/j.biosystemseng.2016.01.017

6. Saleem, M. H., Potgieter, J., & Arif, K. M. (2019). Plant disease detection and classification by deep learning. *Plants, 8*(11), Article 468. DOI: https://doi.org/10.3390/plants8110468

7. Mahlein, A.-K. (2016). Plant disease detection by imaging sensors—Parallels and specific demands for precision agriculture and plant phenotyping. *Plant Disease, 100*(2), 241–251. DOI: https://doi.org/10.1094/PDIS-03-15-0340-FE

8. Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science, 7*, Article 1419. DOI: https://doi.org/10.3389/fpls.2016.01419

9. Hughes, D. P., & Salathé, M. (2015). An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv preprint*, arXiv:1511.08060. URL: https://arxiv.org/abs/1511.08060

10. Too, E. C., Yujian, L., Njuki, S., & Yingchun, L. (2019). A comparative study of fine-tuning deep learning models for plant disease identification. *Computers and Electronics in Agriculture, 161*, 272–279. DOI: https://doi.org/10.1016/j.compag.2018.03.032

11. Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data, 6*, Article 60. DOI: https://doi.org/10.1186/s40537-019-0197-0

12. Piccardi, M. (2004). Background subtraction techniques: A review. In *Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics* (Vol. 4, pp. 3099–3104). IEEE. DOI: https://doi.org/10.1109/ICSMC.2004.1400815

13. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., et al. (2016). TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)* (pp. 265–283). USENIX Association. URL: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi

14. Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint*, arXiv:1704.04861. DOI: https://doi.org/10.48550/arXiv.1704.04861

15. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248–255). IEEE. DOI: https://doi.org/10.1109/CVPR.2009.5206848

16. Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering, 22*(10), 1345–1359. DOI: https://doi.org/10.1109/TKDE.2009.191

17. Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning*. *Proceedings of Machine Learning Research, 28*(3), 1139–1147. URL: https://proceedings.mlr.press/v28/sutskever13.html

18. Loshchilov, I., & Hutter, F. (2017). SGDR: Stochastic gradient descent with warm restarts. In *Proceedings of the 5th International Conference on Learning Representations (ICLR 2017)*. URL: https://openreview.net/forum?id=Skq89Scxx

19. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research, 15*, 1929–1958. URL: https://www.jmlr.org/papers/v15/srivastava14a.html

20. Pongnumkul, S., Chaovalit, P., & Surasvadi, N. (2015). Applications of smartphone-based sensors in agriculture: A systematic review of research. *Journal of Sensors, 2015*, Article 195308. DOI: https://doi.org/10.1155/2015/195308

21. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 770–778). IEEE. DOI: https://doi.org/10.1109/CVPR.2016.90

22. Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297. DOI: https://doi.org/10.1007/BF00994018

23. Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (Vol. 1, pp. 886–893). IEEE. DOI: https://doi.org/10.1109/CVPR.2005.177

24. Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., et al. (2012). Large scale distributed deep networks. In *Advances in Neural Information Processing Systems 25* (pp. 1223–1231). Curran Associates. URL: https://papers.nips.cc/paper/2012/hash/6aca97005c68f1206823815f66102863-Abstract.html

25. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436–444. DOI: https://doi.org/10.1038/nature14539