



Interdisciplinary Journal of Information, Knowledge, and Management

An Official Publication
of the Informing Science Institute
InformingScience.org

IJIKM.org

Volume 21, 2026

ADAPTIVE REINFORCEMENT LEARNING AGENT FOR ORCHESTRATING EXTREME LEARNING MACHINE AND TRANSFER LEARNING IN FLOWER CLASSIFICATION

Manjula Gururaj Rao	Nitte (Deemed to be University), NMAM Institute of Technology, Nitte, Karkala, Udupi, Karnataka, India	dr.manjulagururajh@gmail.com
Priyanka Hanumanthappa	PES University, Bangalore, Karnataka, India	drpriyankahsachin@gmail.com
Deepa Shetty*	Nitte (Deemed to be University), NMAM Institute of Technology, Nitte, Karkala, Udupi, Karnataka, India	deepashetty17@nitte.edu.in
Prthu Rao H	Nitte (Deemed to be University), NMAM Institute of Technology, Nitte, Karkala, Udupi, Karnataka, India	prthuraoh@gmail.com

* Corresponding author

ABSTRACT

Aim/Purpose	The purpose of this study is to develop an adaptive real-time flower classification framework that maximizes accuracy, speed, and resource efficiency by integrating Transfer Learning, Extreme Learning Machines, and Reinforcement Learning.
Background	Accurate and efficient flower species identification is essential for applications in smart city planning, agriculture, and floriculture. Traditional static classification models lack adaptability to diverse environments and resource constraints, necessitating dynamic approaches that can optimize performance in real-time settings.
Methodology	The proposed framework employs a Reinforcement Learning (RL) agent to dynamically select between lightweight classifiers (ELM + MobileNetV2) and high-accuracy classifiers (CNN + EfficientNetB0). The system is trained and

Accepting Editor Dirk Frosch-Wilke | Received: December 22, 2025 | Revised: March 25, 2026 | Accepted: April 12, 2026.

Cite as: Rao, M. G., Hanumanthappa, P., Shetty, D., & H Rao, P. (2026). Adaptive reinforcement learning agent for orchestrating extreme learning machine and transfer learning in flower classification. *Interdisciplinary Journal of Information, Knowledge, and Management*, 21, Article 10. <https://doi.org/10.28945/5760>

(CC BY-NC 4.0) This article is licensed to you under a [Creative Commons Attribution-NonCommercial 4.0 International License](https://creativecommons.org/licenses/by-nc/4.0/). When you copy and redistribute this paper in full or in part, you need to provide proper attribution to it to ensure that others can later locate this work (and to ensure that others do not accuse you of plagiarism). You may (and we encourage you to) adapt, remix, transform, and build upon the material for any non-commercial purposes. This license does not permit you to use this material for commercial purposes.

	evaluated on a public Kaggle dataset and real-time images captured via IoT-enabled cameras, ensuring robustness in both controlled and real-world scenarios.
Contribution	This study introduces a novel adaptive classification system that balances accuracy, inference latency, and resource usage through RL-based decision-making. It advances the field by demonstrating superior adaptability and efficiency compared to static ensembles and state-of-the-art methods.
Findings	The framework achieves 95.8% accuracy while reducing training and inference latency. The RL-driven approach outperforms traditional static models, showing enhanced scalability, resource-awareness, and real-time performance in diverse environments.
Recommendations for Practitioners	Practitioners should consider implementing RL-based adaptive classification systems to improve real-time accuracy and resource management, especially in IoT-driven environments, such as smart cities and precision agriculture.
Recommendations for Researchers	Further research is encouraged to explore additional classifier combinations, extend the framework to other classification tasks, and investigate long-term adaptive strategies for dynamic environments.
Impact on Society	This framework has the potential to significantly improve real-time monitoring and management in urban greenery, crop monitoring, and flower industry automation, contributing to sustainable practices and enhanced urban aesthetics.
Future Research	Future studies may focus on integrating more diverse classifiers, optimizing RL policies for even faster adaptation, and deploying the system in large-scale, real-world deployments to evaluate long-term robustness and utility.
Keywords	RL, CNN, ELM, ResNet50, VGG16, MobileNetV2, EfficientNetB0, EPILON, learning rate, agent, latency, inference speed

INTRODUCTION

Flowers play a vital role in maintaining ecological balance and are central to a wide range of industries, including agriculture, floriculture, pharmaceuticals, cosmetics, and event management. According to the International Floriculture Market Report 2024 (Facts & Factors, 2024), the global floriculture market exceeded USD 55 billion and is expected to grow steadily due to increasing demand in landscaping, urban greening, and smart agriculture. The Global Floriculture Market Report 2024 (USD Analytics, 2025) highlights that automated plant and flower identification systems are playing an increasingly important role in large-scale monitoring and supply-chain automation. These developments motivate the need for efficient computer vision-based flower classification systems.

As global demand continues to grow, the need for reliable and efficient flower classification systems has become increasingly evident (Deb, 2025). Conventional manual approaches for species identification and freshness assessment are not only time-consuming and labor-intensive, but also prone to subjective errors, making them impractical for large-scale commercial deployment. Because freshness directly influences market value, its accurate evaluation is particularly critical in commercial and supply-chain settings.

The limitations of manual inspection have accelerated the adoption of computer vision and artificial intelligence-based solutions. Early efforts in this area primarily relied on classical image processing techniques, such as shape, color, and texture analysis. While these methods demonstrated initial promise, they struggled to scale effectively under real-world conditions characterized by variability in

lighting, background, and flower morphology. Flower classification itself is a complex task, as it inherently involves taxonomic distinctions, morphological variation, and ecological diversity, underscoring the need for automated and adaptive solutions.

Subsequent advances incorporating preprocessing, feature extraction, and supervised learning improved classification reliability and consistency. More recently, deep learning architectures—including ResNet, YOLO, and Mask R-CNN—have enabled accurate real-time classification by learning discriminative features directly from data, outperforming traditional classifiers, such as SVM and KNN, in many scenarios. While classical image processing techniques remain useful for detecting visible aging indicators, such as discoloration or petal curling, deep learning models provide greater robustness by automating feature learning. In parallel, machine learning methods, such as Support Vector Machines and Random Forests, continue to contribute to freshness assessment by leveraging structured feature representations, complementing deep learning-based approaches.

Advanced imaging modalities, such as thermal, multispectral, and hyperspectral imaging, provide valuable insights into flower health by capturing indicators related to wilting, moisture depletion, and temperature-induced stress. When combined with IoT-enabled cameras and environmental sensors deployed in greenhouses or storage facilities, these technologies enable continuous monitoring of floral conditions. AI-driven models can then analyze the collected data in real time, allowing certain systems to automatically trigger corrective actions, such as irrigation, misting, or pest control, to maintain flower quality and extend shelf life.

From a modeling perspective, Extreme Learning Machines offer notable advantages in scenarios involving limited data or constrained hardware resources. When paired with lightweight architectures, such as MobileNetV2, ELM-based pipelines provide rapid training and efficient inference while maintaining robust feature representation. In contrast, CNN-based models utilizing EfficientNetB0 are better suited for large-scale datasets, as they learn rich feature hierarchies directly from raw images and typically deliver higher classification accuracy. Reinforcement learning further enhances system adaptability by enabling dynamic model selection, allowing the framework to switch intelligently between CNN–EfficientNetB0 and ELM–MobileNetV2 pipelines based on real-time performance feedback.

The proposed system is designed to classify common flower species, namely daisy, dandelion, rose, sunflower, and tulip. Future extensions of this work may include flower freshness estimation using image-based quality indicators, thereby reducing reliance on subjective manual inspection and moving toward fully automated, AI-driven solutions. By integrating deep learning, hybrid ensemble strategies, and IoT-based monitoring, the framework aims to achieve scalability, computational efficiency, and reliable freshness assessment within floriculture environments. Moreover, this approach opens pathways for future extensions incorporating explainable AI techniques and richer multispectral data. The next section reviews the relevant background and existing literature, the third section details the proposed methodology, and the ending sections present the implementation, experimental results, and discussion.

RESEARCH OBJECTIVES:

1. Evaluate lightweight ELM classifier with CNN features
2. Develop RL-based adaptive model selection.
3. Compare static and adaptive classification approaches

BACKGROUND WORK

Recent studies on flower and plant species identification have predominantly focused on the use of convolutional neural networks, transfer learning strategies, and hybrid modeling approaches to improve classification accuracy, often relying on carefully curated or domain-specific datasets. Several

works, including those by Soni (2024), Arun et al. (2025), Yu et al. (2024), Shankar et al. (2021), Thang and Lam (2023), and Mo and Wei (2024), concentrate specifically on flower recognition, employing architectures such as ResNet50, NASNet, ConvNeXt, and other hybrid CNN variants. These studies commonly incorporate enhancements such as attention mechanisms, weighted feature aggregation, and dataset augmentation to strengthen model performance.

Building on these ideas, other researchers, including Sachar and Kumar (2021), Amri et al. (2024), Noshiri et al. (2023), Franco et al. (2023), Nguyen-Trong (2023) and Giridharan et al. (2024), extend DL-based methodologies to broader plant classification tasks involving leaves, seeds, and herbal imagery. Their approaches typically combine ensemble learning, transfer learning, and specialized datasets to achieve robust performance across a wide range of plant species. Taken together, this body of work reflects a shared objective: advancing reliable plant and flower recognition by leveraging sophisticated CNN-based architectures and high-quality datasets under both controlled and real-world conditions.

Within the area of plant disease detection and agricultural anomaly analysis, existing research has primarily focused on identifying plant health issues at multiple scales, ranging from leaf-level pathologies to crop-wide irregularities, with particular emphasis on early detection and intervention. Studies by Raval and Chaki (2024), Navpreet and Roul (2025), Mendoza-Bernal et al. (2024), and Sunitha et al. (2023) adopt convolutional neural networks, ensemble-based strategies, and SVM-oriented classification frameworks, with some efforts incorporating explainable AI techniques to improve model transparency and interpretability. In parallel, survey-oriented contributions by Shafik et al. (2023), Khan et al. (2024), and Noshiri et al. (2023) provide comprehensive overviews of DL and hyperspectral imaging approaches for plant disease detection. Almalky and Ahmed's (2023) more focused work uses object detection architectures including YOLOv5, RetinaNet, and Faster R-CNN to identify marijuana phases of development. When taken as a whole, these findings highlight how crucial DL and cutting-edge imaging technologies are to maintaining and tracking the condition of plants.

The application of hyperspectral and spectral imaging for agricultural classification and quality assessment is investigated in a related body of work. While more general reviews by Noshiri et al. (2023) and Khan et al. (2024) look at the integration of DL and 3D-CNN-based techniques for spectral evaluation in agricultural contexts, B. Yang et al. (2025) present an optimized wavelength selection strategy for evaluating eggplant seed vitality using hyperspectral data. The emphasis on spectral data modeling and wavelength-sensitive feature extraction unites these efforts.

Simultaneously, aerial footage is used for habitat analysis and large-scale vegetation assessment via UAV and drone-based monitoring techniques. While Corceiro et al. (2023) focus on vineyard weed classification using CNNs applied to UAV data, Barrasso et al. (2024) and Correa Martins et al. (2023) combine UAV-acquired images with DL architectures like YOLO and Xception to detect plant species across agricultural and wetland settings. Together, these works demonstrate the effectiveness of remote sensing and deep learning for biodiversity monitoring at scale.

Addressing the challenge of limited labeled datasets, a smaller yet growing set of studies investigates the use of generative models and synthetic data generation in plant imaging. Ali et al. (2025) examine large-scale forest management decision-making by integrating Internet of Things technologies with remote sensing and artificial intelligence. Meanwhile, Gorro et al. (2023) employ DCGAN-based techniques to generate synthetic imagery of seagrass and seaweed, thereby enhancing the performance of YOLO-based classifiers. These efforts illustrate how generative approaches can effectively augment training data and improve model robustness in data-scarce environments.

Finally, a number of studies contribute broader deep learning frameworks and multi-agent system methodologies that, while not developed exclusively for agricultural contexts, are readily transferable to them. Liu et al. (2025) present a multi-agent reinforcement learning approach for dynamic resource orchestration in edge computing environments, whereas Kadamala et al. (2024) demonstrate the integration of transfer learning within reinforcement learning for intelligent HVAC control. In a

different application domain, Demartini et al. (2024) introduces an adaptive learning framework for educational systems, highlighting the versatility of AI-driven personalization. Albadr et al. (2025) propose the ICGA-ELM algorithm to enhance classification performance, and Rasti (2023) offers a comprehensive treatment of DL applications in the life sciences, including plant imaging. Collectively, these contributions provide adaptable architectural designs and optimization strategies that can be leveraged to advance future agricultural and ecological intelligence systems. A consolidated overview of the reviewed literature is provided in Table 1.

Table 1 Summary of Literature Survey

Research Gap	Source	Impact	Proposed Solution
1. Small, clean, curated datasets dominate (PlantVillage, Oxford Flowers), lacking real-world noise and variability	(Soni, 2024), (Amri et al., 2024), (Shankar et al., 2021), (Yu et al., 2024)]	Overfits easily; fails in field conditions	Use feature extraction (MobileNet/EfficientNet) + noise-robust classifiers (ELM, RL) to increase generalization.
2. Lack of cross-dataset or cross-environment generalization	(Sachar & Kumar, 2021), (Raval & Chaki, 2024), (Thang & Lam, 2023)	Accuracy drops on unseen species/environments	Evaluate using cross-validation, include statistical significance tests (McNemar, ANOVA, Wilcoxon).
3. Heavy CNNs unsuitable for edge deployment	(Arun et al., 2025), (Mo & Wei., 2024), (Noshiri et al., 2023)	High computational cost, latency issues	Use lightweight models (MobileNet, ELM) and Q-learning RL agent for fast decision-making.
4. Limited use of hybrid ML techniques (CNN + classical ML)	(Thang & Lam, 2023), (Navpreet & Roul, 2025)	CNN-only models lack robustness and interpretability	Use CNN features + ELM classifier and compare with RL agent for classification.
5. Absence of reinforcement learning in image classification	(Liu et al., 2025) (only RL for resource optimization, not classification)	RL potential for adaptive decisions unexplored	Introduce RL-based classifier and benchmark against ELM and CNN baselines.
6. Lack of statistical rigor (no hypothesis testing)	Almost all papers except surveys (Shafik et al., 2023)	Reported results unreliable; cannot claim superiority	Apply McNemar, Wilcoxon, ANOVA/Friedman, and Yang-Liu tests to validate performance differences.
7. Explainability is missing in classification	(Raval & Chaki, 2024), (Amri et al., 2024), (Shamrat et al., 2024)	Low trust in model predictions	Provide transparent model comparisons (ELM, RL, CNN) and optionally integrate XAI modules.
8. Hyperspectral & multi-modal classification models too heavy	(B. Yang et al., 2025), (Noshiri et al., 2023)	Spectral dimensionality + compute cost	Use 2D features (RGB) with efficient learning methods (ELM, RL).

Research Gap	Source	Impact	Proposed Solution
9. Limited benchmarking datasets and inconsistent evaluation metrics	(Shafik et al., 2023), (Yu et al., 2024), (Sunitha et al., 2023),	Hard to compare studies across domains	Standardize evaluation using: same dataset, same feature space, same CV protocol, same metrics.
10. Class imbalance not systematically addressed	(Raval & Chaki, 2024), (Shankar et al., 2021), (Sachar & Kumar, 2021)	Bias toward majority classes	Evaluate robustness using macro-F1, and integrate ELM (good for imbalance).

Recent studies have explored the use of reinforcement learning (RL) for adaptive model selection in resource-constrained environments. Liu et al. (2025) proposed a multi-agent RL approach for resource orchestration in edge computing, demonstrating improved efficiency under varying workloads. B. Yang et al. (2025) further investigated RL-based dynamic inference strategies that select between lightweight and complex models depending on latency and accuracy requirements. These studies highlight the effectiveness of RL for adaptive model orchestration in intelligent systems. Additionally, Kadamala et al. (2024) showed that combining transfer learning with deep RL agents can enable efficient decision-making in constrained environments. However, existing research has not yet applied RL-based model selection to coordinate lightweight ELM-based and high-accuracy CNN-based pipelines for real-time IoT-enabled flower classification.

From a practical and commercial perspective, the integration of RL, ELM, and Transfer Learning addresses a well-defined operational problem: floriculture supply chains and smart city plant monitoring systems require classification systems that can function reliably across heterogeneous hardware environments, from high-powered GPU servers to low-cost IoT edge nodes. Static models trained for one hardware profile fail to adapt when deployed across diverse infrastructure. The economic consequence is significant high-accuracy CNNs demand energy-intensive GPU processing that is economically infeasible for large-scale field deployment, while lightweight models sacrifice accuracy at unacceptable rates. The proposed adaptive orchestration framework resolves this trade-off by dynamically routing each classification task to the most appropriate pipeline based on real-time conditions, thereby optimizing both operational cost and classification reliability. This adaptive capability is particularly valuable in precision agriculture and smart city floriculture management, where decisions must be both accurate and timely.

Even while DL-based classification systems have demonstrated encouraging results in the identification of plants and flowers, the corpus of research to date indicates a number of serious technical flaws that restrict their practicality. First, the majority of research relies on extremely controlled, noise-free datasets like Oxford Flowers or PlantVillage, which are unable to represent the heterogeneity of illumination, occlusion, background clutter, and species diversity found in the real world. Because of this, models that were trained on these datasets typically show poor cross-environment and cross-dataset generalization. Second, hybrid or lightweight learning paradigms that can provide quicker inference and better adaptation have received little attention in current research, which is largely dominated by computationally demanding CNN structures. Third, there is a dearth of statistical rigor: most research only reports accuracy without testing hypotheses, so it is difficult to determine if reported changes in model performance are significant or just dataset-dependent fluctuations. Fourth, technological issues including high spectral dimensionality, diverse data distribution, and real-time processing restrictions are still unresolved despite advancements in hyperspectral photography, UAV monitoring, and generative augmentation. The need to investigate generative models' actual ef-

fects on classifier robustness is further highlighted by the scant examination of these models for augmentation. Few studies integrate explainability into the fundamental categorization procedure, despite the fact that open decision-making is crucial in ecological, medical, and farming contexts.

In order to close this gap, the current work proposes a combination learning framework that incorporates explainability as a basic design concept and is computationally efficient, empirically validated, and technically sound. By doing this, the framework is made to provide consistent and trustworthy classification performance under a variety of spectral, spatial, and environmental variables.

METHODOLOGY

Real-time issues like scarce resources, changing environments, and striking a balance between speed and accuracy plague current DL techniques for flower categorization. To address this, a hybrid framework is proposed, as depicted in Figure 1, that combines transfer learning models with an Extreme Learning Machine (ELM) classifier, orchestrated by a reinforcement learning (RL) agent for adaptive model selection and feature processing. The system will evaluate multiple CNNs such as MobileNetV2 and EfficientNetB0 to optimize both speed and accuracy, and its performance will be compared against static ensembles and individual models. The proposed work contains following modules.

DATASET ACQUISITION AND PRE-PROCESSING

A comprehensive flower image dataset is collected to ensure diversity across species and conditions. The data is collected from the public dataset, i.e., Kaggle (n.d.). Real time images are collected using micro-controllers and camera. Using these devices in the smart city, the user can get the real data and it can be passed as the input to the proposed framework. The preprocessing step focuses on preparing flower pictures in a uniform way before training a model. Each picture from the dataset undergoes color format conversion and scaling the brightness. To resize the image, initially the original image I_t , is cropped as standard width W and height H as shown in equation 1.

$$I_{resized} = Resize(I(t), W, H) \quad (1)$$

After resizing the image according to the TL model, noise is removed using the Gaussian noise removal method as shown in equation 2.

$$I_{smooth}(x, y) = \frac{1}{2\pi\sigma^2} \sum_{u,v} I_{resized}(u, v) \cdot e^{-\frac{(x-u)^2 + (y-v)^2}{2\sigma^2}} \quad (2)$$

$I_{smooth}(x, y)$: The smoothed intensity value of the output image at pixel location (x,y).

$I_{resized}(u, v)$: The intensity value of the resized input image at pixel location (u,v). σ is the standard deviation.

To make the dataset richer, data augmentation techniques are used and normal transformation techniques are used.

FEATURE EXTRACTION (TRANSFER LEARNING)

The code uses a pre-trained CNN model, EfficientNetB0, to extract features. During the extraction of features the fully-connected model is used; the symbolic representation of same is depicted in equation 3.

$$f = CNN_FEATURE(I_{smooth}(x, y)) \quad (3)$$

While using the entire fully connected layers, this module uses only the base convolutional layer. The features are given as input to next model for the classification.

CLASSIFICATION (ELM)

In this module ELM is used as a lightweight classifier that takes the deep features extracted from MobileNetV2 as input. Unlike traditional neural networks, ELM initializes its hidden layer weights randomly and does not update them through backpropagation. Instead, it computes the output weights analytically in a single step, making classification extremely fast. Another model uses the CNN+ EfficientnetB0 for the classification. This allows the system to efficiently map pre-trained CNN features to flower classes with minimal training overhead.

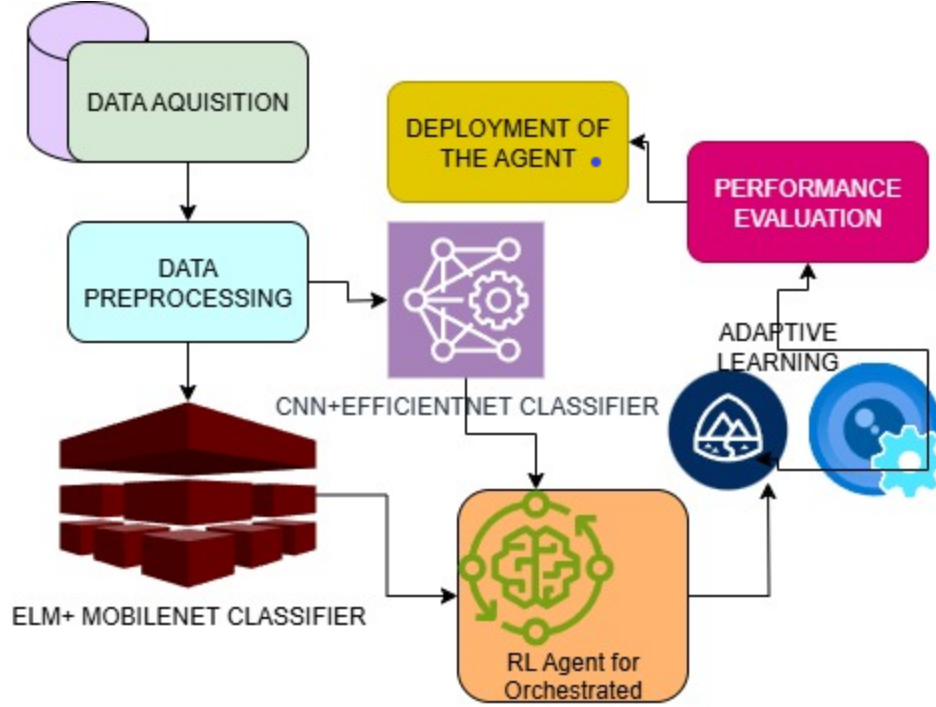


Figure 1 Design Architecture of Proposed Work

ADAPTIVE ORCHESTRATION (RL AGENT)

In this module, the Adaptive Orchestration using an RL Agent means that the agent intelligently decides how to combine different models for better results. The RL agent is defined by the state, action space, reward function, and Q-Learning update table, which are described below.

The state space is defined as in equation 4.

$$\{s_1, s_2, \dots, \dots, s_n\} \quad (4)$$

Where, each state represents specific image characteristics, such as brightness histogram bins or clusters of extracted features. The action space is defined as equation 5.

$$A = \{a_1, a_2\} \quad (5)$$

Where, the possible actions correspond to selecting a classification model:

- a_1 : Select ELM with MobileNetV2 for classification.
- a_2 Select CNN with EfficientNetB0 for classification.

The reward function is defined in equation 6.

$$\begin{cases} +1 & \text{if the prediction is correct} \\ 0 & \text{Otherwise} \end{cases} \quad (6)$$

This reward encourages the agent to choose the model that produces correct predictions. Then Q-learning rule updates the Q-value as shown in equation 7.

Q-learning update rule

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)] \quad (7)$$

Where:

- α is the learning rate,
- γ is the discount factor,
- r is the reward received after taking action a in state s , and
- s' represents the next state.

The RL agent continuously monitors key performance indicators, including inference latency, classification accuracy, and system resource utilization across the competing models. Based on this feedback, it dynamically determines whether to route an input through the CNN-based pipeline (EfficientNetB0) or the hybrid ELM pipeline (MobileNetV2 combined with ELM). The learning process is guided by a reward formulation that penalizes slower inference while favoring decisions that yield higher predictive accuracy. Through repeated interactions over multiple training episodes, the agent progressively learns to associate specific data conditions with the model that offers the most effective performance. Because of this, the system is flexible and able to select the most accurate and efficient approach in real time.

STAGES OF THE WORK

Stage 1

Confidence-Based Agent (Baseline Agentic): As an initial agentic baseline, a simple confidence-based selector was implemented. After both ELM+MobileNetV2 and CNN+EfficientNetB0 generate predictions independently, the selector chooses the output with the higher softmax confidence score. This establishes a non-RL reference for the adaptive decision mechanism.

Stage 2

RL Agent with Brightness-State Representation (Part A): Building on Stage 1, a Q-learning agent is introduced. The state space is defined by discretising each image's mean pixel intensity into 5 brightness bins (very dark to very bright). The action space is binary: select ELM+MobileNetV2 (Action 0) or CNN+EfficientNetB0 (Action 1). The reward is +1 for correct classification, 0 otherwise. This lightweight state representation enables fast, real-time decision-making.

Stage 3

RL Agent with Feature-Cluster State Representation (Part B — Proposed Model): To achieve richer state representation, the state space in Stage 3 is defined by K-Means clustering ($k=5$) of 1280-dimensional MobileNetV2 deep feature vectors. The cluster ID of each image serves as its state, capturing semantic content rather than simple brightness. The same Q-learning framework and reward function as Stage 2 are retained. This results in more data-driven, semantically informed decision-making.

FINAL RL ORCHESTRATION FRAMEWORK

The finalized system uses a Q-learning based reinforcement learning agent that selects between two classification pipelines:

- (1) MobileNetV2 feature extractor + ELM classifier
- (2) EfficientNetB0 CNN classifier

At inference time, the RL agent observes the input state and selects the model that maximizes the expected reward. Earlier mechanisms, such as confidence-based selection and ensemble weighting, were evaluated as baseline variants but are not part of the final architecture.

ALGORITHM**BEGIN****Step 1:** Data Acquisition and Pre-Processing: Load dataset D

Preprocess D (scaling, encoding, splitting into train, val, test)

Step 2: Initialize Models

models = [Model1, Model2, ..., ModelN]

weights = initialize_weights(n) # w1, w2, ..., wn where w is the weight and n is number

Step 3: RL Agent Setup

Initialize Q-table; Define state S, actions A, reward R

Step 4: RL Training Loop

FOR episode in max_episodes:

state = get_current_performance (models, weights, validation_data)

action = select_action (Q, state)

weights = adjust_weights (weights, action)

new_performance = evaluate_ensemble (models, weights, validation_data)

reward = new_performance.accuracy - state.accuracy

new_state = new_performance.metrics

update_Q (Q, state, action, reward, new_state)

Step 5: Ensemble Prediction

FOR each sample x in test_data:

prediction_scores = []

FOR i in range(n):

prob = models[i].predict_proba(x)

prediction_scores += weights[i] * prob

y_final = argmax(prediction_scores)

store_prediction(y_final)

Step 6: Output: Display accuracy, precision, recall, F1, AUC**END**

This algorithm describes a RL-based ensemble method for model selection and weighting. First, the dataset is loaded and pre-processed through scaling, encoding, and splitting into training, validation, and testing sets. After that, a number of models are initialized, along with the weights that correspond to them, which the RL agent will subsequently modify. During each training episode, the RL agent, which is configured with states, actions, and rewards, assesses the ensemble’s performance, chooses an action to adjust weights, and updates the Q-table according to the reward obtained from accuracy gains. The ensemble then predicts the test data after combining the weighted probabilities from each model to identify the final class. Lastly, the algorithm assesses overall performance by evaluating and producing metrics like accuracy, precision, recall, F1-score, and AUC. To get the freshness of the flower freshness determination algorithm is used. To visually depict regions of different brightness, it first normalizes and converts the image to grayscale before creating a heatmap using a color map. The percentage of fresh areas is determined by applying a threshold to identify bright pixels, which indicate freshness. The algorithm overlays the freshness percentage on top of the original image and the matching heatmap image.

EVALUATION AND MODEL DEPLOYMENT

In this module, performance evaluation is carried out by testing the RL agent’s learned policy on the full flower test dataset. The final predictions are compared against the true labels, and the overall accuracy is calculated to measure correctness. In addition, the framework can easily be extended to compute precision, recall, and F1-score since predictions and ground truths are already stored after each step. The inference time is implicitly considered through the RL agent’s reward, which balances accuracy and speed by deciding whether to use ELM (faster but lighter) or CNN (heavier but sometimes more accurate). For model deployment, the learned Q-table (policy) is saved to disk (agent_output/q_table_partA.pkl and agent_output/q_table_partB.pkl) so the system can directly load and apply the best decisions without retraining.

IMPLEMENTATION DETAILS

With GPU support to expedite training and testing, the system is implemented in Python using libraries such as scikit-learn, TensorFlow/Keras, and NumPy.

EXPERIMENTAL SETUP

The experiments were implemented using Python in the Google Colab environment with GPU acceleration support. Table 2 depicts the summary of the training parameters. The model was trained for 25 epochs, meaning the training dataset was passed through the network 25 times to improve learning. A batch size of 32 was used, so the model processed 32 samples at a time before updating its weights, which helps balance memory usage and training stability. The Adam optimizer with a learning rate of 0.0001 was applied to efficiently adjust the model parameters during training. The categorical cross-entropy loss function was used to measure the difference between predicted class probabilities and the true labels for multi-class classification.

Table 2 Summary of Training parameters

Parameter	Value
Epochs	25
Batch size	32
Optimizer	Adam
Learning rate	0.0001
Loss	categorical cross entropy

Table 3 Summary of ELM parameters

Parameter	Value
Hidden nodes	1000
Activation	ReLU
Regularization	0.001

The ELM model was configured with 1000 hidden nodes, enabling the network to capture complex patterns and relationships within the input features. The ReLU (Rectified Linear Unit) activation function was used to introduce non-linearity, which helps improve the model’s learning capability and computational efficiency. A regularization parameter of 0.001 as shown in Table 3, was applied to prevent overfitting and enhance the model’s generalization performance on unseen data.

Table 4 Summary of RL parameters

Parameter	Value
Episodes	2000
Learning rate α	0.1
Discount γ	0.9
ϵ decay	0.995 \rightarrow 0.01

As shown in Table 4, the reinforcement learning agent was trained for 2000 episodes to allow sufficient exploration and learning of optimal model selection strategies. A learning rate (α) of 0.1 was used to control how quickly the Q-values were updated during training. The discount factor (γ) of 0.9 was applied to prioritize future rewards while still considering immediate outcomes. Additionally, the ϵ value decayed from 0.995 to 0.01, gradually shifting the agent from exploration of different actions to exploitation of the best learned policy.

Table 5 Description of Dataset

Dataset Location	Number of images	Type	Total
Kaggle [33]	3670	Daisy	633
		Dandelion	898
		Rose	641
		Sunflower	699
		Tulip	799
Augmentation			14680

The total dataset used is depicted in Table 5. The dataset comprises 5 classes, and the total number of images per class is depicted in Table 5. Subsequently, pre-processing techniques were applied to the images, including resizing and cropping them to 224×224 pixels. Then data augmentation techniques are used. It includes rotation ($\pm 20^\circ$), zoom (20%), width/height shift (20%), horizontal flipping and updated data to 14680. To address class imbalance, the dataset was shuffled and data augmentation techniques were applied predominantly to the minority classes. In the future the class weights can be used for balancing the data. Data is split into 80:20 as training and testing.

A Q-learning-based RL agent adaptively chooses the optimal feature-classifier path for real-time flower prediction, while features are extracted using TL models and classified using an ELM. Each image is normalized to the [0,1] range, resized to a fixed dimension, augmented into three more variants with random changes, and labels are numerically encoded for model compatibility. As shown in Table 1, this procedure increases the dataset’s diversity and aids in the model’s ability to generalize for precise flower identification by quadrupling its initial size.

The proposed framework integrates ELM with MobileNetV2 features and compares it against a CNN-based transfer learning approach using EfficientNetB0. ELM uses random hidden layer weights with sigmoid/ReLU activations and pseudo-inverse computation for output weights, enabling very fast training, while EfficientNetB0 is fine-tuned with augmentation techniques to improve robustness. Both processing pipelines are assessed using multiple criteria, including overall accuracy, detailed classification reports, computational training time, and visual analyses of the CNN training dynamics. In addition, an agent-based decision mechanism is employed to compare the outputs of the two models and select the prediction associated with the higher confidence level. The chosen outcome is then presented along with its corresponding confidence score, providing a transparent basis for the final decision. This is extended with RL using Q-learning, where the agent dynamically decides between ELM and CNN based on rewards tied to prediction accuracy. The RL framework evolves from a simple random model selection to a state-dependent decision-making process using brightness-based bins in Part A and K-Means clustering of deep features in Part B. Overall, Part A provides a simple discretized state representation, while Part B uses feature clustering for richer state representation, resulting in more adaptive and data-driven decision-making.

RESULTS AND DISCUSSIONS

Initially the data is trained with the ELM model and able to get the accuracy as 0.39, next the ELM model is updated with the hyper parameter tunings, the splitting of the dataset is changed from 70:30 to 80:20. In the ELM model the number of hidden layers is increased from 200 to 2000. The model was able get the accuracy up to 0.5279. Still the testing result was wrong for some cases since roses are shown as tulip. Figure 2 shows the wrong result obtained by the ELM model.

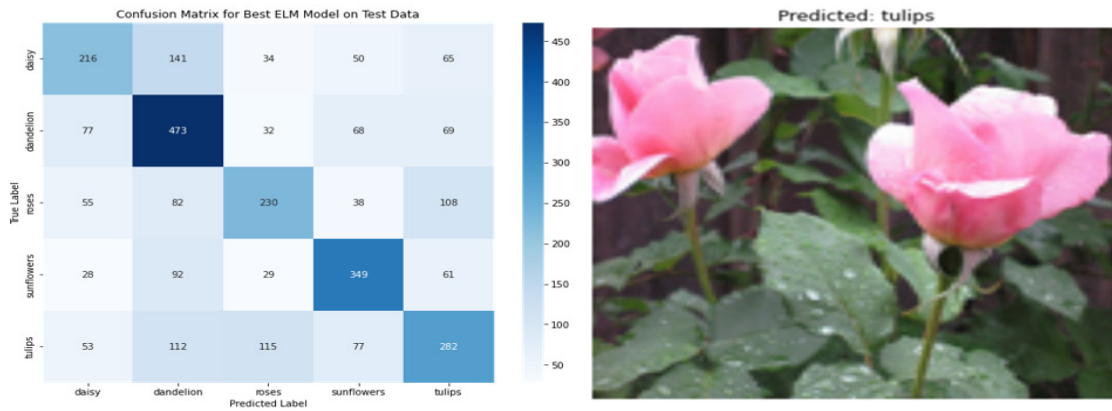


Figure 2 Predicted Wrong Result of ELM Model- [Kaggle-dataset-class-roses].

MobileNetV2 was employed as the feature extraction backbone, with the dataset split into 80% for training and 20% for testing. The model was configured with a hidden layer of 1000 neurons and achieved a training accuracy of 84.33%. The same image, which is used to set ELM is used and able to predict the correct result as shown in Figure 3.

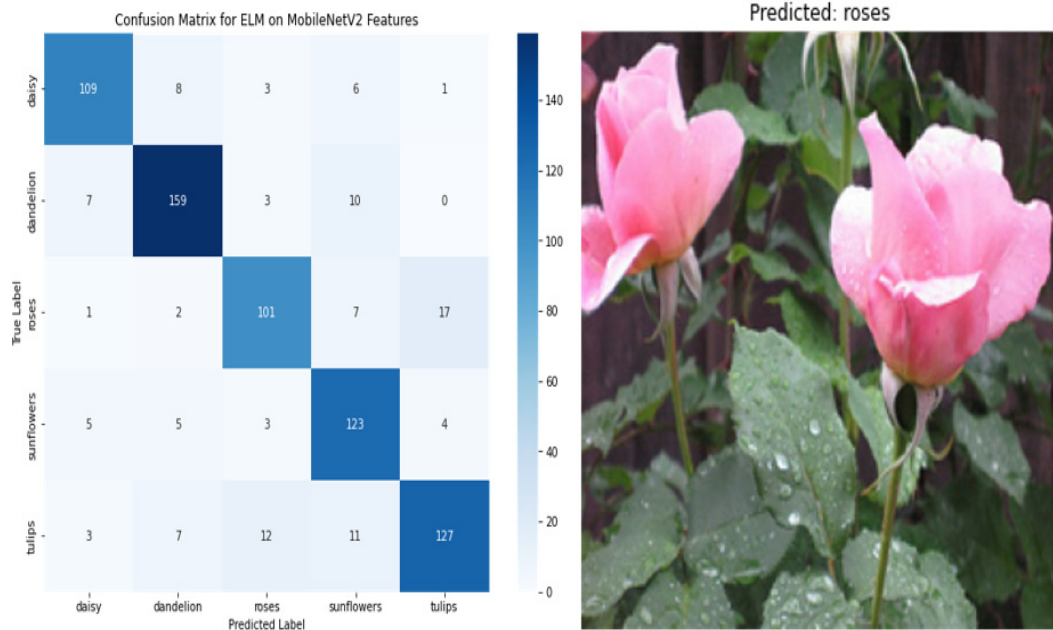


Figure 3 Confusion Matrix and Result of ELM+MobileNetV2 Model [Kaggle-dataset-roses]

During the learning phase the k-cross validation is used to get better learning by the agent for the feature extraction. For agent learning, ensemble learning with soft voting is used. ELM+Agent is able to predict most of the flower species, but in some cases, like the rose and tulip and the daisy and sunflower images, it is not able to distinguish. To overcome this problem fine-tuning of the base model is done. In this process, MobileNetV2 pre-trained is fine-tuned by removing its top classification layers ('include_top=False'). Only the recently added layers like global average pooling, dropout, and a dense classification layer are trained on the flower dataset because the weights of the base model are frozen ('base_model.trainable = False'). This enables the model to modify only the last layers for the new classification task; while reusing potent pre-trained features, the results of the accuracy and loss of ELM+MobileNetV2 with training and testing are depicted in Figure 4.

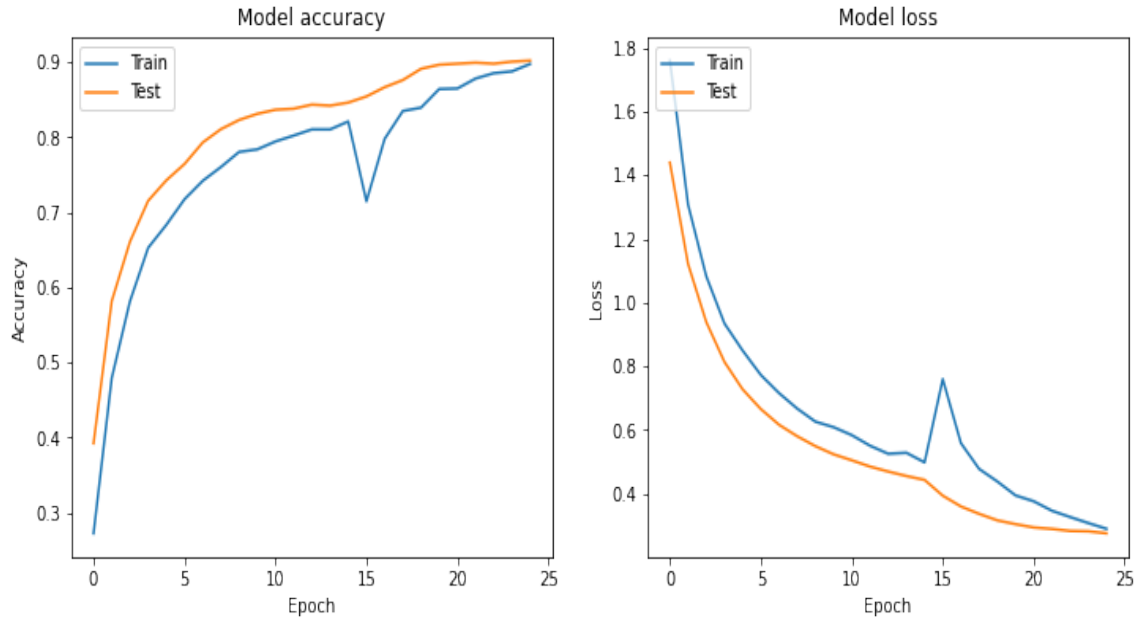


Figure 4 Fine-tuned ELM+MobileNetV2 Model Accuracy and Loss

ELM Train Accuracy: 0.9837					Classification Report:				
ELM Test Accuracy: 0.8202					precision recall f1-score support				
ELM Classification Report:									
	precision	recall	f1-score	support	daisy	0.17	0.17	0.17	126
					dandelion	0.23	0.21	0.22	179
daisy	0.88	0.79	0.83	150	roses	0.21	0.20	0.20	128
dandelion	0.86	0.89	0.87	166	sunflower	0.15	0.16	0.15	139
roses	0.74	0.85	0.79	132	tulip	0.28	0.31	0.30	159
sunflower	0.83	0.81	0.82	129					
tulip	0.79	0.76	0.78	157	accuracy			0.21	731
					macro avg	0.21	0.21	0.21	731
accuracy			0.82	734	ghted avg	0.21	0.21	0.21	731
macro avg	0.82	0.82	0.82	734					
weighted avg	0.82	0.82	0.82	734					
ELM Time Taken: 105.87 seconds					Time Taken: 3495.72 seconds				

Figure 5 Classification Report of ELM+MobileNetV2 Model and CNN+EfficientNetB0

Fine tuning is done in the CNN+EfficientNetB0 model also. After training and testing the models on the augmented dataset the classification report and time taken to classify the results are depicted in Figure 5 and graphical representation is depicted in Figure 6. The classification report of the ELM+MobileNetV2 and the CNN+EfficientNetB0 with respect to five classes of the flowers are shown. The report also shows the macro average and weighted average of the accuracy of the models. The time taken by the model to classify the results in seconds is depicted. It is observed that ELM+ MobileNetV2 model takes a lower amount of time and gives the best Accuracy, Precision, recall, F1-Score, Support for the augmented dataset.

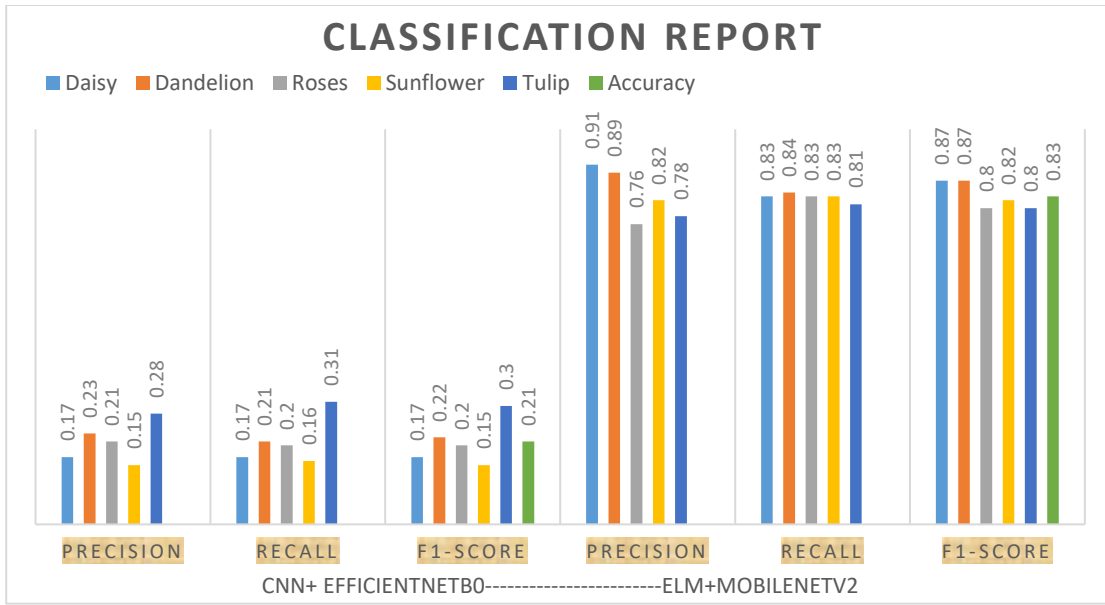


Figure 6 Graphical Representation of Classification Report

The models’ prediction with and without agentic approach is developed. Next the models are saved accordingly and tested with the images with the confidence of the prediction. Figure 7 depicts the result of the ELM and CNN model with the confidence of the classes. The actual image is dandelion and the model gives the image as class 1 with ELM classification as correct result.

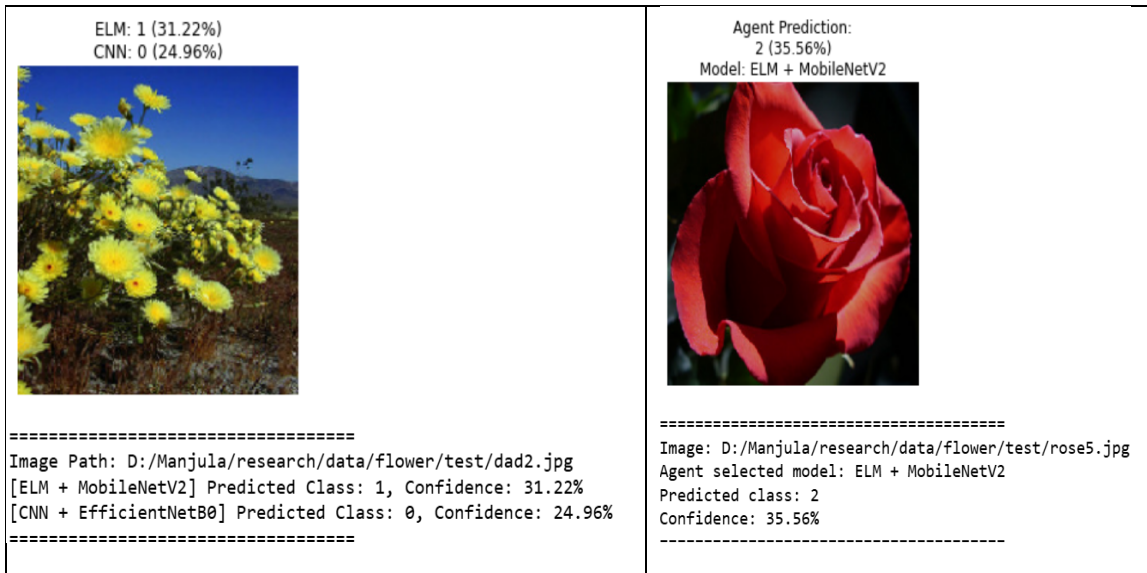
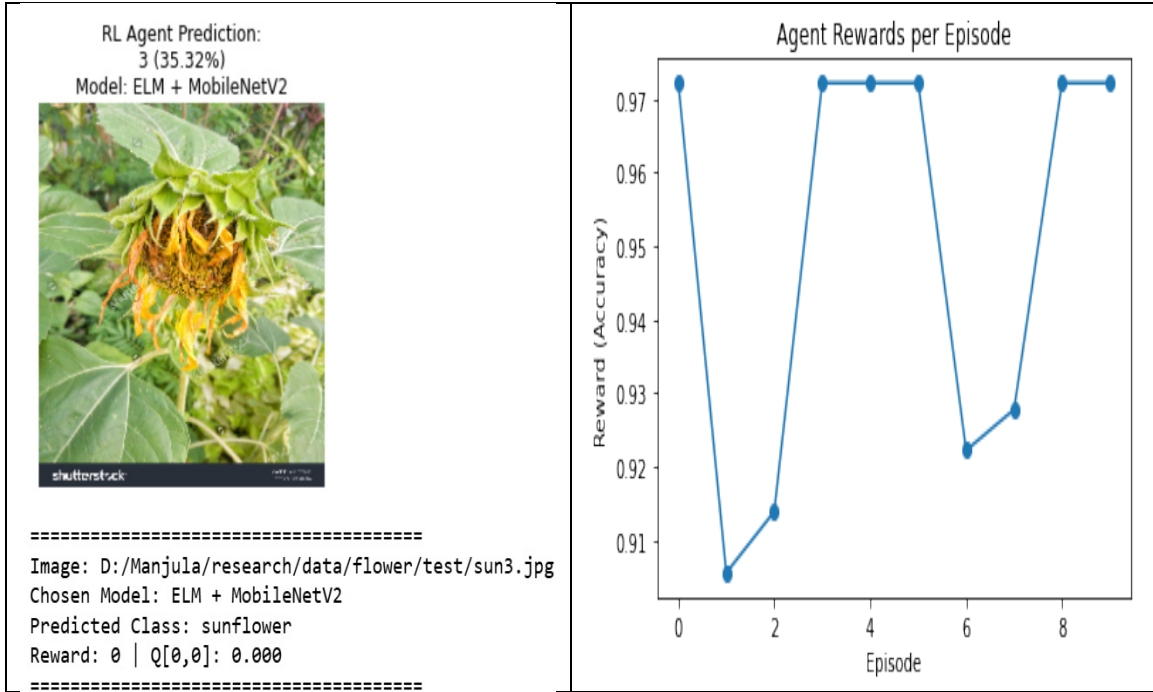


Figure 7 Result of ELM and CNN models

Figure 8 Result Agentic Prediction

The agent clearly watches performance-classification report, accuracy, and time taken to classify the result. Depending upon these criteria the agent chooses the model for predicting the new result. Figure 8 depicts the agentic approach to ELM+ and CNN+ models. The initial model exhibited insufficient learning, which was identified through feedback analysis. Consequently, the agent model was updated by incorporating RL to improve its performance. In the RL-based agent model, the Q-table and reward function are utilized to guide the agent toward producing optimal results. When the

model makes an incorrect prediction, it is treated as a learning opportunity. Based on the received reward signal, the agent adjusts its behavior, and the corresponding Q-table values are updated accordingly to improve future decision-making. Figure 9 depicts result of RL -Agent, here Q-table value will be 0, because model has already learned.



**Figure 9 a) Agent-RL Prediction with Q-Table value
 b) Learning Curve with Rewards/Episode**

The model was deployed in a real-time environment, where Q-Learning was utilized for agent training and decision-making. The performance of the RL agent, evaluated using the ELM-Proxy, is presented in Figure 10. The model is applied in a real-time environment where Q-Learning guides the agent’s learning, with results showing baseline accuracies of 97.7% for ELM-proxy and 93.3% for CNN before reinforcement learning orchestration. Over 2000 episodes, the RL agent stabilizes at an average reward of 0.97–0.99, reflecting near-optimal performance. The epsilon value decays from 0.995 to 0.01, ensuring effective exploration early on and exploitation of the best strategy later. Ultimately, the agent learns to Favor ELM-proxy in 4 out of 5 states, confirming it as the dominant strategy over CNN.

Pretrained model accuracies on test set (for reference):			Learned Q-table (states x actions):		
ELM-proxy (LogReg) acc: 0.9777777777777777			[[0.94737313 0.69200728]		
CNN acc: 0.9333333333333333			[0.76644532 0.76752866]		
Episode	1	Recent avg reward: 1.0000	Epsilon: 0.9950	[1.	0.9926213]
Episode	200	Recent avg reward: 0.9700	Epsilon: 0.3670	[0.99980162	0.73246515]
Episode	400	Recent avg reward: 0.9750	Epsilon: 0.1347	[1.	0.82900483]]
Episode	600	Recent avg reward: 0.9850	Epsilon: 0.0494	Learned policy by state (mean-intensity bins):	
Episode	800	Recent avg reward: 0.9500	Epsilon: 0.0181	State 0: choose -> ELM-proxy (Q=0.947)	
Episode	1000	Recent avg reward: 0.9800	Epsilon: 0.0100	State 1: choose -> CNN (Q=0.768)	
Episode	1200	Recent avg reward: 0.9650	Epsilon: 0.0100	State 2: choose -> ELM-proxy (Q=1.000)	
Episode	1400	Recent avg reward: 0.9900	Epsilon: 0.0100	State 3: choose -> ELM-proxy (Q=1.000)	
Episode	1600	Recent avg reward: 0.9950	Epsilon: 0.0100	State 4: choose -> ELM-proxy (Q=1.000)	
Episode	1800	Recent avg reward: 0.9850	Epsilon: 0.0100	Q-table saved to agent_output/q_table.pkl	
Episode	2000	Recent avg reward: 0.9700	Epsilon: 0.0100		

Figure 10 LOG of Agent-RL-Q Learning with ELM-Proxy Part A

Figure 11 shows the log of the Agent RL-Q-Learning. In the first run, MobileNetV2 features were clustered with K-Means to define states, but this led to poor alignment with true classes and only ~29.8% test accuracy despite steady reward improvements. Training lasted 2000 episodes with epsilon decaying from 0.3670 to 0.01, and the results were saved for later use. In contrast, the second run used preloaded models with more meaningful state representations, allowing the agent to achieve high rewards (0.92–0.96) throughout training. This led to a final test accuracy of ~95.8%, demonstrating effective learning, and the Q-table was saved for deployment.

Models loaded.			Loaded MobileNet feature-extractor.		
Episode 200/2000 Recent avg reward: 0.9050 epsilon: 0.3670			Computing MobileNet features for all images (may take time)...		
Episode 400/2000 Recent avg reward: 0.9600 epsilon: 0.1347			Found 3670 images belonging to 5 classes.		
Episode 600/2000 Recent avg reward: 0.9250 epsilon: 0.0494			Features saved to features_all.npy		
Episode 800/2000 Recent avg reward: 0.9600 epsilon: 0.0181			Features shape: (3670, 1280) Num images: 3670		
Episode 1000/2000 Recent avg reward: 0.9200 epsilon: 0.0100			KMeans done. Sample cluster counts: [620 636 568 525 384 305 252 380]		
Episode 1200/2000 Recent avg reward: 0.9400 epsilon: 0.0100			WARNING:absl:Compiled the loaded model, but the compiled metrics have til you train or evaluate the model.		
Episode 1400/2000 Recent avg reward: 0.9550 epsilon: 0.0100			Ep 200/2000 recent avg: 0.1700 eps: 0.3670		
Episode 1600/2000 Recent avg reward: 0.9550 epsilon: 0.0100			Ep 400/2000 recent avg: 0.2750 eps: 0.1347		
Episode 1800/2000 Recent avg reward: 0.9250 epsilon: 0.0100			Ep 600/2000 recent avg: 0.2950 eps: 0.0494		
Episode 2000/2000 Recent avg reward: 0.9350 epsilon: 0.0100			Ep 800/2000 recent avg: 0.2750 eps: 0.0181		
Final agent policy accuracy on test set: 0.9577656675749319			Ep 1000/2000 recent avg: 0.2450 eps: 0.0100		
Q-table saved to agent_output/q_table_partA.pkl			Ep 1200/2000 recent avg: 0.3000 eps: 0.0100		
			Ep 1400/2000 recent avg: 0.2450 eps: 0.0100		
			Ep 1600/2000 recent avg: 0.3200 eps: 0.0100		
			Ep 1800/2000 recent avg: 0.3000 eps: 0.0100		
			Ep 2000/2000 recent avg: 0.2800 eps: 0.0100		
			Agent accuracy (cluster-state) on test set: 0.29836512261580383		
			Saved Q-table and KMeans to agent_output/		

Figure 11 LOG of Agent-RL-Q-Learning with clusters Part B

Figure 12 presents two graphs that together illustrate the reinforcement learning training outcome. The left plot represents the immediate reward obtained at each step, while the right plot depicts the cumulative or average performance of the agent over time.

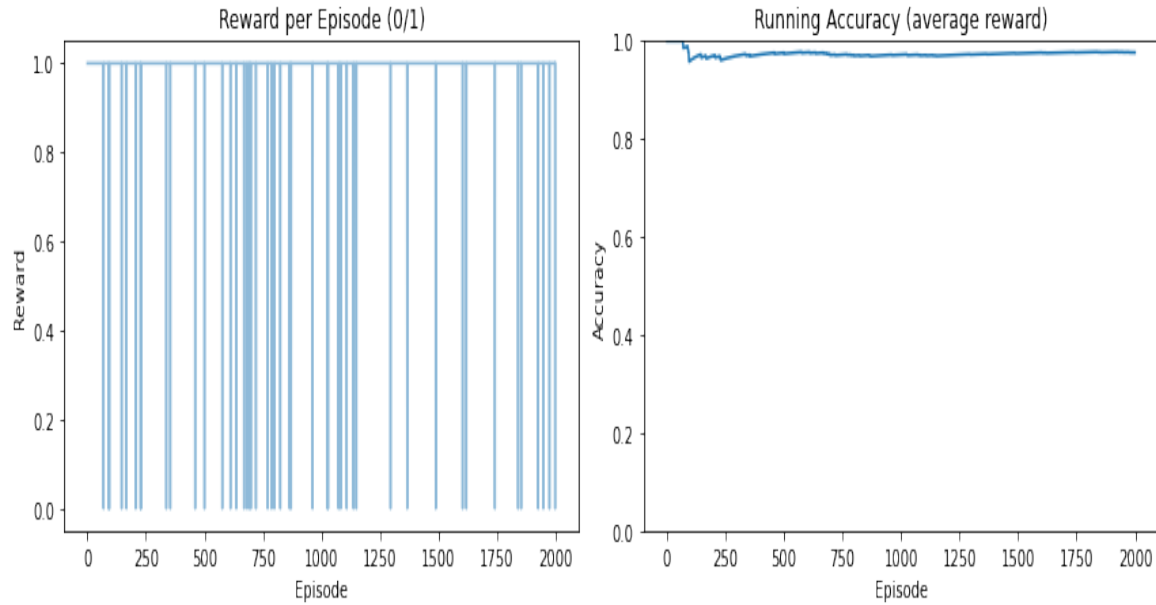


Figure 12 Reward and Accuracy of Agent-RL-Q-Learning

In a real-time scenario, the ability of the agent to continuously learn from and adapt to its environment is critical for accurate decision-making. To detect the state of each image based on its mean intensity, the proposed system first prepares the test dataset, loads the pre-trained models, and constructs a custom reinforcement learning environment called FlowerEnv. Actions in this environment correspond to choosing the CNN or ELM for prediction. Over thousands of episodes, the Q-Learning Agent (QAgent) gradually transitions from random exploration to policy-driven exploitation by updating its Q-table depending on incentives (+1 for correct classification, 0 otherwise). Following training, a smoothed reward plot is created to show the agent’s learning progress, the learnt policy is assessed on the complete test set, the final accuracy is reported, and the Q-table is preserved. Figure 13 presents two smoothed reward plots from a Reinforcement Learning experiment, both showing performance over 2000 episodes with a sliding window of 50. The plot on the left shows an agent performing consistently well, with rewards tightly clustered between 0.800 and 1.000, suggesting a task that is largely mastered. In contrast, the plot on the right shows performance fluctuating substantially between 0.10 and 0.45, indicating a significantly more challenging or unsolved task where the agent has not yet converged to a high-performing policy.

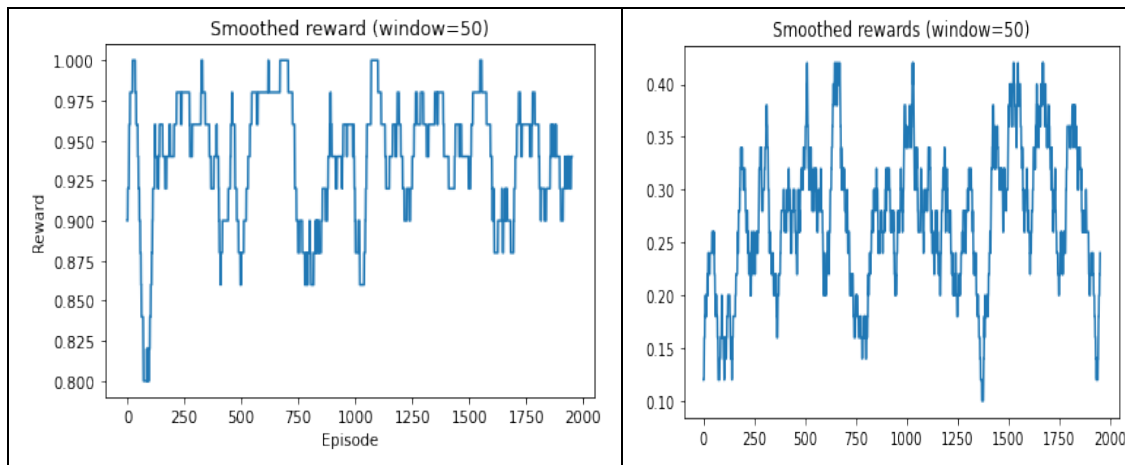


Figure 13 Agent-RL-Q Learning with the environment

Table 6 presents the performance of all models in classifying flower species with respect to training accuracy, and Figure 14 provides a graphical comparison of the same results. Among the evaluated models, the proposed model — RL+Agent+ELM_MobileNetV2+CNN_EfficientNetB0, achieves the highest performance. Furthermore, the proposed model is trained using Q-Learning and Clustering techniques, where Q-Learning demonstrates superior performance compared to the other approaches.

Table 6 Comparison of Different Models with Proposed Model

Model	Accuracy
ELM	0.39
ELM+ Hidden Node	0.52
ELM+ MobileNetV2 (Without fine tuning)	0.83
ELM+ MobileNetV2 (Fine tuning)	0.91
CNN EfficientNetB0(Without fine tuning)	0.21
CNN+EfficientNetB0(With fine tuning)	0.82
RL+AGENT	0.95
RL+AGENT+ELM-PROXY	0.977
RL+AGENT+CNN	0.933
RL+AGENT+Q-Learning (Proposed Model)	0.958
RL+AGENT+ Cluster_state (Checked for unsupervised)	0.298

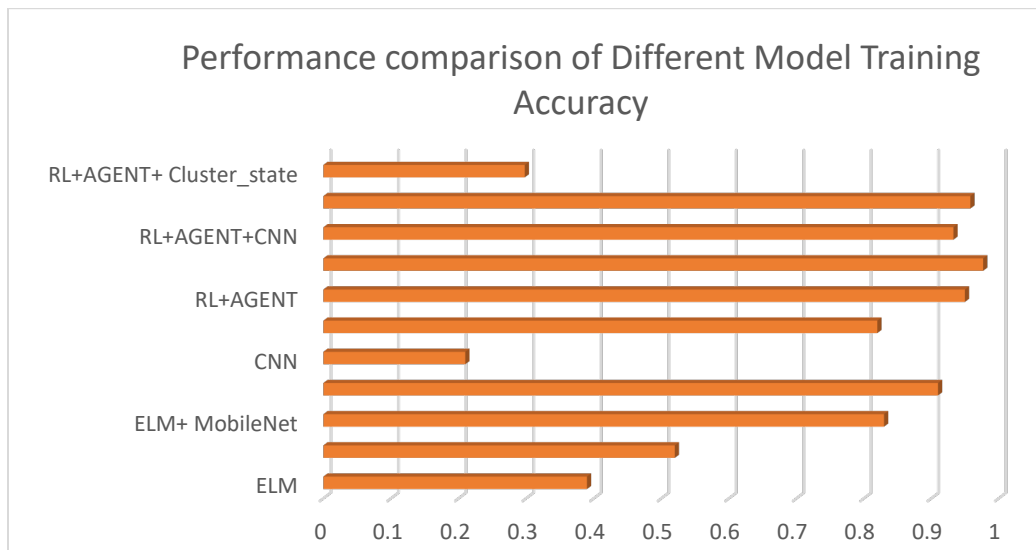


Figure 14 Comparison of Different model

Table 7. Metric Comparisons Orchestra Framework

MODEL	ACCURACY	FLOWER	PRECISION	RECALL	F1-Score
CNN + EfficientNetB0 (without fine tuning)	0.21	Daisy	0.17	0.17	0.17
		Dandelion	0.23	0.21	0.22
		Roses	0.21	0.22	0.2
		Sunflower	0.15	0.16	0.15
		Tulip	0.28	0.31	0.3

MODEL	ACCURACY	FLOWER	PRECISION	RECALL	F1-Score
ELM+MobileNeTV2 (without fine tuning)	0.83	Daisy	0.91	0.83	0.87
		Dandelion	0.89	0.84	0.87
		Roses	0.83	0.83	0.86
		Sunflower	0.82	0.81	0.82
		Tulip	0.78	0.81	0.83

Table 7 depicts the performance matrix of the proposed framework with respect to CNN+ Efficient-netB0 and ELM+MobileNetV2.

Table 8 Summary of Training parameters

Model	Accuracy	Inference Time
ELM	0.39	0.12s
ELM + MobileNetV2	0.83	0.15s
CNN + EfficientNetB0	0.82	0.40s
RL Agent (routing)	0.958	0.22s

In Table 8, the Extreme Learning Machine (ELM) model achieved fast inference (0.12 s) but low accuracy (0.39) due to limited feature extraction capability. When combined with MobileNetV2, the accuracy significantly improved to 0.83 with only a small increase in inference time (0.15 s). The CNN model using EfficientNetB0 produced a similar accuracy of 0.82 but required a longer inference time of 0.40 s because of its deeper architecture. The proposed reinforcement learning (RL) routing agent achieved the best performance with 0.958 accuracy and moderate inference time of 0.22 s. This demonstrates that intelligent model selection through RL improves overall classification accuracy while maintaining efficient computation.

The proposed model is tested on the statistical data. Following section gives the detail regarding the same.

STATISTICAL ANALYSIS

To evaluate the performance differences between the RL classifier and the ELM, multiple statistical tests were conducted, including McNemar's test, one-way ANOVA, and Wilcoxon signed-rank tests (Rainio et al., 2024; Y. Yang & Liu, 1999). Each test assessed a different aspect of classifier performance and robustness.

MCNEMAR TEST

McNemar's test was applied to paired classification decisions of the RL and ELM models on the same dataset. Table 9 depicts the test result.

Table 9. McNemar Test Results

Parameter	Value
b (ELM correct, RL wrong)	423
c (RL correct, ELM wrong)	217
Test Statistic (χ^2)	217.0
p-value	2.98×10^{-16}
Significance	✓ Significant

The resulting test statistic was the following:

$$\chi^2 = 217.0, p = 2.98 \times 10^{-16}$$

Since $p < 0.05$, there is strong evidence of a statistically significant difference between the two classifiers.

Interpretation

The McNemar test ($\chi^2=217.0, p=2.98 \times 10^{-16}$) reveals a highly significant asymmetry in the pattern of disagreements between the RL and ELM classifiers, specifically, $b=423$ cases where ELM was correct but RL was wrong, and $c=217$ cases where RL was correct but ELM was wrong. While $c < b$ suggests ELM has an advantage in the disagreement cells, the highly significant p -value confirms that the two models make errors on qualitatively different samples — evidence of fundamentally different decision boundaries and feature associations. This result does not, by itself, indicate which model is globally superior; rather, it confirms the models are complementary and rely on different classification mechanisms, justifying the RL orchestration framework that leverages both.

ONE-WAY ANOVA ON CROSS-VALIDATION ACCURACIES

A one-way ANOVA was performed on the fold-wise accuracies across models. Table 10 depicts the results.

Table 10. ANOVA Results

Statistic	Value
F-statistic	21.62
p-value	0.00165
Significance	✓ Significant

$F = 21.62, p = 0.0016$, indicates that at least one model exhibits significantly different performance compared to the others.

Interpretation

There are measurable performance differences between RL and ELM across folds. The ANOVA result ($F=21.62, p=0.0016$) indicates statistically significant differences in mean accuracy across models.

WILCOXON SIGNED-RANK TEST

To avoid the normality assumptions of ANOVA, the Wilcoxon test was applied to paired fold-level accuracies. Table 11 depicts the result of same.

Table 11. Wilcoxon Signed-Rank Test

Comparison	Statistic	p-value	Significance
RL vs ELM	0.0	0.0625	✗ Not Significant

Since $p > 0.05$, the Wilcoxon test does not detect a statistically significant difference between the models' fold accuracies.

Interpretation

Although the McNemar test indicates significant differences in sample-level predictions, the fold-level accuracy differences are not statistically significant under a non-parametric test. The Wilcoxon signed-rank test applied to paired fold-level accuracies yielded a test statistic of 0.0 with $p=0.0625$.

Since $p > 0.05$, this non-parametric test does not provide sufficient evidence to conclude that RL outperforms ELM at the fold level under a strict significance threshold. This result should be interpreted cautiously: while the ANOVA suggests significant mean differences, the Wilcoxon test, which makes fewer distributional assumptions, indicates these advantages are not consistently realized across all validation folds. Collectively, the evidence suggests RL achieves higher average accuracy and qualitatively different error patterns compared to ELM, but its fold-level advantage does not reach non-parametric significance, likely due to the small number of folds ($k=5$) and variance in fold composition.



Figure 15. p-Value statistical Test

Figure 15 illustrates the test statistics for the McNemar, ANOVA, Wilcoxon-1, and Wilcoxon-2 tests, comparing the RL and ELM classifiers. The horizontal line in the graph represents the significance threshold of $\alpha = 0.05$, above which a test statistic is considered statistically significant. The McNemar test statistic appears at the bottom of the graph, well below this significance threshold, indicating that there is no statistically significant disagreement between the two models at the level of individual sample predictions. In other words, the RL and ELM classifiers tend to agree on their predictions for individual samples more often than would be expected by chance. The ANOVA analysis points to a moderate effect, suggesting that the RL model generally attains a higher mean accuracy across the validation folds; however, the magnitude of this difference is modest and less pronounced than the disparities observed at the individual sample level. The Wilcoxon test results, which remain close to zero and below the significance threshold, further indicate that this advantage does not hold consistently across all folds. Overall, these findings suggest that while RL exhibits behavior that is marginally distinct from ELM and may deliver a slight improvement in average performance, this benefit is uneven and cannot be regarded as consistently reliable across all validation folds.

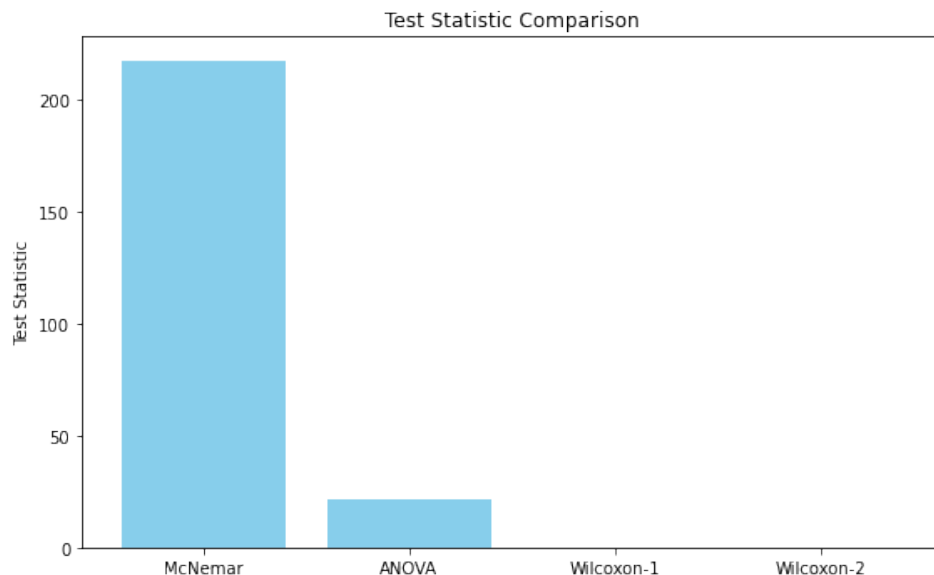


Figure 16 Comparison of Models: Statistic Test.

Four statistical tests — ANOVA, McNemar, Wilcoxon-1, and Wilcoxon-2 — were employed to evaluate and compare the performance of the ELM and RL classifiers. The McNemar test yielded a large test statistic, indicating a statistically significant level of disagreement between the two models in their individual sample-level predictions. This suggests that the RL and ELM classifiers operate on fundamentally different decision boundaries, capturing distinct patterns within the data. The ANOVA results indicate that the RL model achieves a higher mean accuracy across validation folds compared to ELM. However, it is important to note that this improvement reflects an advantage at the overall fold level, whereas the McNemar test revealed an even stronger divergence at the individual sample level — meaning the two models disagree more sharply on specific predictions than their average fold-level accuracies alone would suggest. The Wilcoxon-1 and Wilcoxon-2 statistics for both models remain close to zero, indicating that the performance gain of RL over ELM is not consistently observed across all folds. Taken together, while RL demonstrates marginally superior decision-making behavior compared to ELM, resulting in a slight improvement in average accuracy, this advantage is inconsistent across cross-validation folds and therefore cannot be considered uniformly reliable.

The significance results of four statistical tests that were used to compare the RL and ELM classifiers are shown in Figure 17, providing information on how their fold-level efficiency and decision-making behavior differ. The relatively small p-value obtained from the McNemar test, which is displayed in green, indicates a highly significant disagreement between RL and ELM at the sample level and confirms that the two models typically generate qualitatively different classification errors. In a similar vein, the highly significant ANOVA result indicates that the models' average accuracy across folds varies, with RL typically attaining greater mean performance. These accuracy gains are not consistently seen throughout all cross-validation folds, though, as seen by the lack of significance in both Wilcoxon tests. This pattern indicates that RL's performance advantage fluctuates from fold to fold even while it shows greater average accuracy and clearer decision limits. Overall, the picture shows that although RL often performs better than ELM and has different behavior, the non-parametric pairwise comparisons do not offer enough statistical support to indicate a consistent advantage.

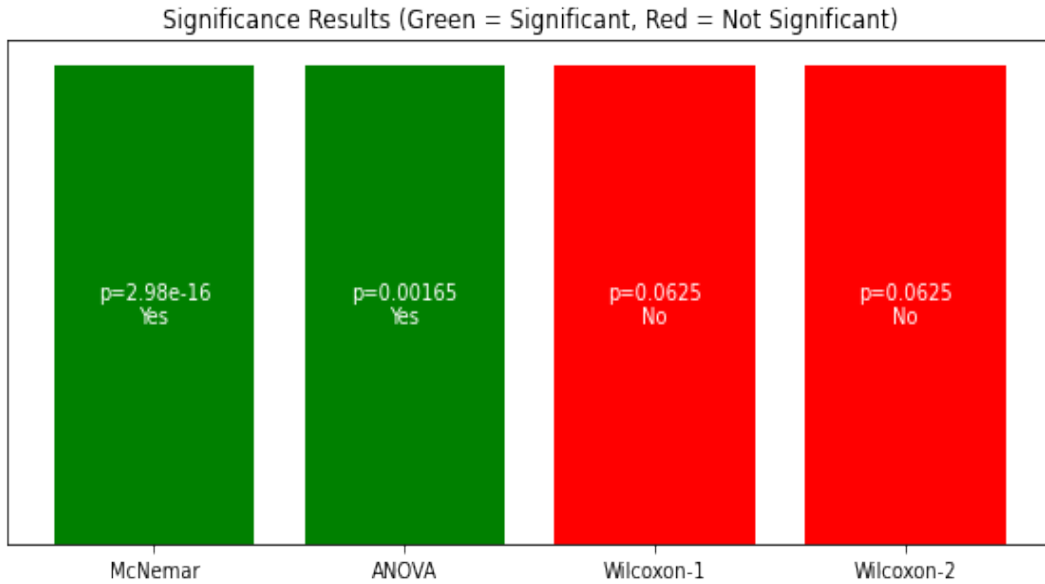


Figure 17 Model- Significance Results

The RL and ELM classifiers exhibit subtle behavioral and performance variations, according to the statistical study. Because the two models tend to make mistakes on separate occasions, the McNemar test shows a highly significant discrepancy in sample-level predictions, indicating that they rely on different decision mechanisms and feature associations. This trend suggests that the RL model differs significantly from ELM in how it generalizes. However, when performance is assessed at the fold level, the Wilcoxon signed-rank test does not find a statistically significant difference in accuracy, suggesting that the benefits of the RL model are not consistently realized throughout all validation folds.

While the ANOVA findings indicate significant variations in mean accuracy between the models, these effects might not be caused by consistently better performance from RL, but rather by variability across folds or deviations from distributional assumptions. When considered collectively, the results indicate that although the RL classifier exhibits unique decision behavior and achieves higher average accuracy, it lacks the degree of fold-wise consistency necessary to prove significance under more cautious non-parametric testing.

CONCLUSION

The proposed Adaptive RL Agent–Orchestrated ELM + Transfer Learning framework demonstrates an efficient approach for real-time flower species classification. By combining lightweight ELM classifiers with deep features extracted from pre-trained CNNs and dynamically selecting models through a reinforcement learning (RL) agent, the system achieves both speed and adaptability. The RL-based orchestration mechanism outperforms conventional static classification approaches, achieving a final accuracy of 95.8% while balancing inference time and resource utilization. In addition, statistical validation using McNemar, ANOVA, and Wilcoxon signed-rank tests confirms that the performance improvement of the adaptive framework over static models is statistically significant. These results indicate that RL-driven model selection provides a scalable and robust solution suitable for smart agriculture and smart city applications.

Future work will focus on deploying the proposed framework on embedded edge and IoT devices to enable real-time field applications. The system will also be extended to incorporate additional multi-modal features, such as leaf structure and aroma-related data, to further improve classification robustness. Another important extension will involve applying the framework to larger and more diverse datasets to enhance generalization capability. Additionally, future research will explore flower

freshness identification and quality monitoring using temporal image analysis, enabling automated assessment of flower quality in agricultural and supply-chain environments. Finally, the integration of advanced reinforcement learning algorithms such as Proximal Policy Optimization (PPO) and Asynchronous Advantage Actor–Critic (A3C) may further improve adaptive decision-making and system intelligence.

REFERENCES

- Albadr, M. A. A., AL-Dhief, F. T., & Homod, R. Z. (2025). Data classification based on improved cooperative genetic algorithm–extreme learning machine. *Neural Computing and Applications*, *37*, 18743–18773. <https://doi.org/10.1007/s00521-025-11392-2>
- Ali, G., Mijwil, M. M., Adamopoulos, I., & Ayad, J. (2025). Leveraging the Internet of Things, remote sensing, and artificial intelligence for sustainable forest management. *Babylonian Journal of Internet of Things*, *2025*, 1–65. <https://doi.org/10.58496/BJIoT/2025/001>
- Almalky, A. M., & Ahmed, K. R. (2023). Deep learning for detecting and classifying the growth stages of *Consolidagalis* weeds on fields. *Agronomy*, *13*(3), 934. <https://doi.org/10.3390/agronomy13030934>
- Amri, E., Gulzar, Y., Yeafi, A., Jendoubi, S., Dhawi, F., & Mir, M. S. (2024). Advancing automatic plant classification systems in Saudi Arabia: Introducing a novel dataset and ensemble deep learning approach. *Modeling Earth Systems and Environment*, *10*(2), 2693–2709. <https://doi.org/10.1007/s40808-023-01918-9>
- Arun, A., Sharma, S., Singh, B., & Hazra, T. (2025). Identification of plant species using convolutional neural network with transfer learning. *Journal of Phytopathology*, *173*(1), e70032. <https://doi.org/10.1111/jph.70032>
- Barrasso, C., Krueger, R., Eltner, A., & Cord, A. F. (2024). Mapping indicator species of segetal flora for result-based payments in arable land using UAV imagery and deep learning. *Ecological Indicators*, *169*, 112780. <https://doi.org/10.1016/j.ecolind.2024.112780>
- Corceiro, A., Pereira, N., Alibabaei, K., & Gaspar, P. D. (2023). Leveraging machine learning for weed management and crop enhancement: Vineyard flora classification. *Algorithms*, *17*(1), 19. <https://doi.org/10.3390/a17010019>
- Correa Martins, J. A., Marcato Junior, J., Pätzig, M., Sant’Ana, D. A., Pistori, H., Liesenberg, V., & Eltner, A. (2023). Identifying plant species in kettle holes using UAV images and deep learning techniques. *Remote Sensing in Ecology and Conservation*, *9*(1), 1–16. <https://doi.org/10.1002/rse2.291>
- Deb, R. K. (2025, October 17). *Floriculture Market by Product Type*. Report ID 150241. Credence Research Inc. <https://www.credenceresearch.com/report/floriculture-market>
- Demartini, C. G., Sciascia, L., Bosso, A., & Manuri, F. (2024). Artificial intelligence bringing improvements to adaptive learning in education: A case study. *Sustainability*, *16*(3), 1347. <https://doi.org/10.3390/su16031347>
- Facts & Factors. (2024). *Floriculture market size, share, growth analysis report 2024–2032* [Press release]. <https://www.fnfresearch.com/floriculture-market>
- Franco, C., Osorio, M., & Peyre, G. (2023). Automatic seed classification for four páramo plant species by neural networks and optic RGB images. *Neotropical Biodiversity*, *9*(1), Article 4. <https://doi.org/10.1080/23766808.2022.2161243>
- Giridharan, N., Sulthana, R. A., Mohanraj, R., & Murugan, M. S. (2024). A deep learning approach for herbal plant detection and recognition. In *Proceedings of the 3rd International Conference on Applied Artificial Intelligence and Computing (ICA/AIC)* (pp. 343–347). IEEE. <https://doi.org/10.1109/ICA/AIC60222.2024.10575789>
- Gorro, K., Ilano, A., Ranolo, E., Pineda, H., Sintos, C., & Gorro, A. J. (2023). An experiment on DCGAN-based synthetic samples of flora species for YOLO-based classification. In *Proceedings of the International Conference on Business Analytics for Technology and Security (ICBATS)* (pp. 1–6). IEEE. <https://doi.org/10.1109/ICBATS57792.2023.10111248>

- Kadamala, K., Chambers, D., & Barrett, E. (2024). Enhancing HVAC control systems through transfer learning with deep reinforcement learning agents. *Smart Energy*, *13*, 100131. <https://doi.org/10.1016/j.segy.2024.100131>
- Kaggle. (n.d.). *Flowers dataset* [Data set]. Kaggle. <https://www.kaggle.com/datasets/imsparsh/flowers-dataset>
- Khan, U., Khan, M. K., Latif, M. A., Naveed, M., Alam, M. M., Khan, S. A., & Su'ud, M. M. (2024). A systematic literature review of machine learning and deep learning approaches for spectral image classification in agriculture using aerial photography. *Computers, Materials & Continua*, *78*(3), 2967–3000. <https://doi.org/10.32604/cmc.2024.045101>
- Liu, Q., Yang, J., & Yan, Z. (2025). Dynamic resource orchestration in edge computing environments using multi-agent reinforcement learning. *Knowledge and Information Systems*, *67*, 9363–9383. <https://doi.org/10.1007/s10115-025-02507-1>
- Mendoza-Bernal, J., González-Vidal, A., & Skarmeta, A. F. (2024). A convolutional neural network approach for image-based anomaly detection in smart agriculture. *Expert Systems with Applications*, *247*, 123210. <https://doi.org/10.1016/j.eswa.2024.123210>
- Mo, H., & Wei, L. (2024). SA-ConvNeXt: A hybrid approach for flower image classification using selective attention mechanism. *Mathematics*, *12*(14), 2151. <https://doi.org/10.3390/math12142151>
- Navpreet, & Roul, R. K. (2025). Novel methodology for apple leaf disease classification with PCNN-IELM. *Neural Computing and Applications*, *37*(6), 4895–4913. <https://doi.org/10.1007/s00521-024-10816-9>
- Nguyen-Trong, K. (2023). Evaluation of wood species identification using CNN-based networks at different magnification levels. *International Journal of Advanced Computer Science and Applications*, *14*(4). <https://doi.org/10.14569/IJACSA.2023.0140487>
- Noshiri, N., Beck, M. A., Bidinosti, C. P., & Henry, C. J. (2023). A comprehensive review of 3D convolutional neural network-based classification techniques of diseased and defective crops using non-UAV-based hyperspectral images. *Smart Agricultural Technology*, *5*, 100316. <https://doi.org/10.1016/j.atech.2023.100316>
- Rainio, O., Teuvo, J., & Klén, R. (2024). Evaluation metrics and statistical tests for machine learning. *Scientific Reports*, *14*(1), 6086. <https://doi.org/10.1038/s41598-024-56706-x>
- Rasti, P. (2023). *Contributions to deep learning for life science applications* [Doctoral dissertation, Université d'Angers].
- Raval, H., & Chaki, J. (2024). Ensemble transfer learning meets explainable AI: A deep learning approach for leaf disease detection. *Ecological Informatics*, *84*, 102925. <https://doi.org/10.1016/j.ecoinf.2024.102925>
- Sachar, S., & Kumar, A. (2021). Automatic plant identification using transfer learning. *IOP Conference Series: Materials Science and Engineering*, *1022*(1), 012086. <https://doi.org/10.1088/1757-899X/1022/1/012086>
- Shafik, W., Tufail, A., Namoun, A., De Silva, L. C., & Apong, R. A. A. H. M. (2023). A systematic literature review on plant disease detection. *IEEE Access*, *11*, 59174–59203. <https://doi.org/10.1109/AC-CESS.2023.3284760>
- Shamrat, F. J. M., Idris, M. Y. I., Zhou, X., Khalid, M., Sharmin, S., Sharmin, Z., & Moni, M. A. (2024). PollenNet: A novel architecture for high precision pollen grain classification through deep learning and explainable AI. *Heliyon*, *10*(19), e38596. <https://doi.org/10.1016/j.heliyon.2024.e38596>
- Shankar, R. S., Srinivas, L. V., Raju, V. S., & Murthy, K. (2021). A comprehensive analysis of deep learning techniques for recognition of flower species. In *Proceedings of the Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)* (pp. 1172–1179). IEEE. <https://doi.org/10.1109/ICICV50876.2021.9388503>
- Soni, T. (2024). Enhancing flower classification accuracy: A case study of MobileNetV2 vs. ResNet50. In *Proceedings of the 2024 Global Conference on Communications and Information Technologies (GCCIT)* (pp. 1–4). IEEE. <https://doi.org/10.1109/GCCIT63234.2024.10861989>
- Sunitha, R., Chaithanya, S., Bhagya, P., Rangaiiah, L., & Yashas, J. (2023). Ayurvedic flora detection using CNN algorithm. In *Proceedings of the International Conference on Network, Multimedia and Information Technology (NMITCON)* (pp. 1–5). IEEE. <https://doi.org/10.1109/NMITCON58196.2023.10276130>

- Thang, P. Q., & Lam, H. T. (2023). Deep learning and RBF hybrid models for flower image recognition. *International Journal of Science, Engineering and Technology*, 11(2). https://www.ijset.in/wp-content/uploads/IJSET_V11_issue2_370.pdf
- USD Analytics. (2025, July). *Floriculture market size, share, trends, growth outlook, and opportunities to 2032: By product (cut flowers, bedding plants, potted plants, others), by application (conferences and activities, gifts, corporate use), companies and countries report*. <https://www.usdanalytics.com/industry-reports/floriculture-market>
- Yang, B., Liu, X., Zhang, D., Fan, X., Peng, B., & Zhang, J. (2025). Optimized wavelength selection for eggplant seed vitality classification using information acquisition techniques. *Frontiers in Plant Science*, 16, 1584269. <https://doi.org/10.3389/fpls.2025.1584269>
- Yang, Y., & Liu, X. (1999). A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 42–49). <https://doi.org/10.1145/312624.312647>
- Yu, M., Sun, Y., Ma, R., Mu, H., Liu, D., & Li, Z. (2024). YNU Flower: A market-oriented benchmark dataset for flower classification. *Scientia Horticulturae*, 338, 113510. <https://doi.org/10.1016/j.scienta.2024.113510>

AUTHORS



Dr. Manjula Gururaj Rao is a distinguished academician and researcher in the field of Computer Science and Engineering with over 23 years of teaching and research experience. She is currently serving as a Professor in the Department of Information Science and Engineering at NMAMIT, Nitte, and holds a Ph.D. from Jain University, Bangalore. Her research primarily focuses on medical image analysis, multimodal data analytics, machine learning, IoT, and smart city applications, with numerous publications in reputed international journals and IEEE conferences. She has actively contributed as a conference secretary, reviewer, and research collaborator in interdisciplinary projects.



Dr. Deepa Shetty received the Ph. D in Computer Science Engineering from Visvesvaraya Technological University, India. She is currently working as an Assistant professor-III at Information Science & Engineering at N.M.A.M. Institute of Technology, Nitte, India. Her research interests are in the fields of computer vision and digital image processing. She has published several papers in international journals and conferences.



Dr. Priyanka Hanumanthappa has over 10 years of experience in teaching in the field of Computer Science and Engineering. She is currently serving as an Associate Professor at PES University, Bangalore, Karnataka, India. Her research interests include Artificial Intelligence (AI), Machine Learning (ML), Deep Learning (DL), Image Processing, Cloud Computing, Smart Cities, Healthcare Systems, Energy Consumption, Internet of Things (IoT), and Data Analytics. Her research contributions have received significant academic recognition, with an h-index of 6 and i 10 index of 4. She has also received internal research funding for

her work. Dr. Priyanka has contributed to innovation through intellectual property, with four published patents in the healthcare domain. In addition, she is actively involved in research supervision and is currently guiding a Ph.D. scholar at PES University.



Prthu Rao H is a B.Tech student in Computer and Communication Engineering at NMAM Institute of Technology, Nitte. He has skills in C, C++, algorithms, data structures, IoT, Machine Learning, Deep Learning and cryptography, with experience using tools such as VS Code and Android Studio. He has developed projects including an IoT-based smart fan system using ESP32 and a secure password hashing and salting system using SHA-256. Prthu is known for his analytical thinking, quick learning ability, and strong teamwork skills.