DEEP LEARNING-BASED FLOWER CLASSIFICATION

Estellyn Hoo Weay, Lee Tian Xin, Ho Xin Yong Cinee, Chan Kah Kei

Faculty of Computing and Informatics, Multimedia University, Malaysia

ABSTRACT

Flower classification using computer vision faces challenges due to lighting, background, and angle variations. This project aims to develop robust algorithms for accurate flower classification. We employed deep learning models: DenseNet-121, InceptionV3, and a custom Convolutional Neutral Network (CNN), leveraging transfer learning for DenseNet-121 and InceptionV3. Optimization was done using Adam and Stochastic Gradient Descent (SGD), with performance measured by accuracy, precision, recall, F1-score, confusion matrices, prediction time, and parameter size. Our experiments show that DenseNet-121 optimized with Adam achieved the highest accuracy (95%), outperforming other models. InceptionV3 also performed well across different optimizers, while the custom CNN was less effective, particularly with SGD. Misclassifications mainly involved Lily, often confused with Tulip due to similar petal shapes. Grad-CAM visualizations highlighted inconsistent focus areas as a source of these errors.

Index Terms— Convolutional Neural Networks, DenseNet-121, InceptionV3, Adam, SGD, flower classification

1. INTRODUCTION

The classification of various flower types using computer vision represents a complex problem in computer vision due to the inherent variability in images, such as lighting conditions, diverse backgrounds, variations in color, and different angles from which the flowers are photographed. These variations make it difficult for traditional methods to reliably and accurately classify flower types.

One of the primary motivations for this project is to contribute to advancements in computer vision. The challenges faced in flower classification necessitate the development of more robust algorithms. The improvements in algorithms derived from this project will potentially have far-reaching implications beyond making flower classification faster and easier, enhancing the broader capabilities of computer vision systems in critical fields such as medical imaging and autonomous vehicles.

Another significant motivation is to improve agricultural applications. Accurate flower type classification can greatly benefit crop management, pest control, and pollination strategies. By leveraging this technology, farmers can make more informed decisions, leading to increased crop yields and more efficient resource allocation. This reduces reliance on manual identification, minimizes human error, and lowers labor costs, thereby promoting more sustainable and profitable agricultural practices.

A further motivation is to support research and conservation efforts. Accurately identifying different flower species is crucial for mapping their geographic distribution, pinpointing areas rich in biodiversity, and monitoring changes in plant populations over time. This information can aid in assessing ecosystem health and developing effective conversation strategies.

The primary objectives of this project are as follows:

- To develop a deep learning model that can accurately and reliably classify different types of flowers despite variations in lighting, background, color, and angle.
- To evaluate and compare the performance of deep learning models on flower classification and identify the most effective model based on various performance metrics.

2. RELATED WORK

Flower classification poses a significant challenge in computer vision due to the wide array of flower classes that share similar shapes, appearances, or background elements. Traditional methods often struggle to reliably and accurately classify flower types under these conditions due to reliance on features extracted from segmented flower regions only, limiting their ability to capture the full spectrum of variability found in flower images.

Early approaches to flower classification primarily relied on traditional computer vision methods such as feature extraction and clustering. Various features were extracted, including Scale-Invariant Feature Transform (SIFT), Speeded-Up Robust Features (SURF), Hue Saturation Value (HSV), RGB color model, and Color Texture Moments (CTM). The features were then organized into a codebook using k-means clustering to form a dictionary of visual words. Each image was represented in a simplified bag-of-features model, which is a histogram of the clustered features. Classification was performed using Support Vector Machines (SVM) with a one-versus-all strategy, where the classifier learned to categorize each image into one of the predefined

classes based on its feature histogram. During testing, the same feature extraction and combination techniques were applied to new images, and the SVM classifier classified these images based on their feature combinations. Although the combined SURF+CTM feature set offered enhanced performance compared to other feature combinations, achieving an accuracy of 74.59%, this level of accuracy prompted further exploration into more advanced techniques [1].

The introduction of deep learning marked a significant advancement in the accuracy and robustness of flower classification systems. Mete and Ensari (2019) proposed a classification system leveraging Deep CNN and data augmentation to improve flower classification performance. Their approach involved using the Inception-v3 model for feature extraction, followed by classification using various machine learning algorithms, including SVM, Random Forest, KNN, and Multi-Layer Perceptron (MLP). By augmenting the data through vertical flipping, they addressed the challenge of limited labeled data, achieving remarkable accuracy rates: 98.5% for the Oxford-102 Flowers dataset using an SVM classifier and 99.8% for the Oxford-17 Flowers dataset with an MLP classifier. This study demonstrated the efficacy of combining deep learning for feature extraction with traditional machine learning for classification [2].

Further advancements were made with the development of specialized deep learning models tailored for flower classification. Qin et al. (2019) introduced an improved Convolutional Neural Network model (A-LDCNN) that incorporates an attention mechanism and a Linear Discriminant Loss Function (LD-loss). Their model used the VGG-16 network pre-trained on ImageNet to perform feature learning on preprocessed flower images. The attention feature was constructed by fusing local features from multiple intermediate convolution layers with global features from the fully connected layer, enhancing the model's focus on key areas of the image. The introduction of LD-loss aimed to minimize intra-class feature distance while maximizing inter-class feature distance, addressing the challenges of inter-class similarity and intra-class variability in flower image classification. Their experiments on the Oxford-102 flower dataset achieved an accuracy of 87.6%, highlighting the model's ability to accurately recognize flowers under natural conditions [3].

He et al. (2021) proposed a Multi-scale Dense Residual Network (MSDRNet) to address the limitations of traditional residual networks, such as insufficient feature utilization and limited convolution scale. MSDRNet utilized multi-scale convolutions and dense connections to enhance feature flow and utilization. The model achieved an accuracy of 87.91% on the tf-flowers dataset, outperforming other models like DenseNet-121, ResNet-50, Inceptionv3, and VGG-19. This approach demonstrated the potential of advanced network architectures in improving classification accuracy [4].

Pawar et al. (2022) presented a practical application of CNNs for flower classification, focusing on real-time identification through a web application. Their system utilized TensorFlow to implement a CNN model capable of identifying various flower species. The model achieved an accuracy of 98.68%, showcasing its potential as a valuable tool for botanists, travelers, and environmental enthusiasts [5].

The progression from traditional machine learning techniques to sophisticated deep learning models has significantly enhanced the accuracy and effectiveness of flower classification systems. Traditional methods, while foundational, were limited by their reliance on manual feature extraction and segmentation. The emergence of deep learning, particularly CNNs, and their integration with attention mechanisms, custom loss functions, and multi-scale dense networks, has substantially improved classification performance. These advancements highlight the transformative impact of deep learning on flower classification, paving the way for further innovations and practical applications in the field.

3. APPROACH

3.1. Convolutional Neural Networks

CNNs are commonly used in the field of image data as they excel in computer vision tasks such as image recognition and image classification as well as object detection. Hence, CNNs are extensively employed in artificial intelligence to develop image classifiers.

CNNs are composed of three key layers: convolutional layers, pooling layers, and fully connected layers. Convolutional layers apply filters to the input data to detect various features. Pooling layers reduce the dimensionality of the feature maps, preserving essential information while decreasing computational load. Fully connected layers integrate the features identified in the earlier layers to classify the images accurately.

CNNs outperform traditional neural networks in image processing due to their ability to use convolutional layers with fewer parameters and employ parameter sharing. In contrast, traditional networks, consisting solely of fully connected layers, have difficulty processing spatial data and tend to overfit.

3.2 DenseNet-121

DenseNet-121 is a specific type of DenseNet with 121 layers. DenseNet, or Densely Connected Convolutional Network, is a neural network architecture that enhances feature propagation and reuse through dense connections. In DenseNet, each layer receives inputs from all preceding layers and passes its output to all subsequent layers, promoting efficient training and strong gradient flow. This

connectivity pattern reduces the number of parameters and mitigates the vanishing gradient problem, leading to improved performance and parameter efficiency compared to traditional networks.

DenseNet's concatenation of feature maps from previous layers instead of summation is represented by the equation:

$$x_l = H_l([x_0, x_1, \ldots, x_{l-1}])$$

where x_l is the output feature of the l-th layer, H_l is the transformation function (typically batch normalization, ReLU, and convolution operations) applied at the l-th layer, $[x_0, x_1, \ldots, x_{l-1}]$ is the concatenation of feature maps from all previous layers.

3.3 InceptionV3

Unlike most popular CNNs that rely on stacking deeper convolutional layers to enhance performance, the Inception network employs a complex, heavily engineered architecture to optimize speed and accuracy.

InceptionV3 builds upon the foundational Inception modules, which capture information at multiple scales through parallel convolutions of varying filter sizes. This modular design allows InceptionV3 to effectively learn complex patterns in images while maintaining computational efficiency.

Key improvements include factorized convolutions, the Root Mean Square Propagation (RMSProp) optimizer, BatchNorm in the auxiliary classifiers, and label smoothing to prevent overfitting.

3.4 Optimization Algorithms

Optimization algorithms are essential for training neural networks for image classification, minimizing the loss function and enhancing accuracy.

3.4.1. Adam

Adam optimization algorithm was developed after merging the strengths of Momentum and RMSProp optimization algorithms.

The Adam algorithm adjusts the learning rates dynamically for each parameter by leveraging the first and second moments of the gradients: the first moment captures the average gradient, while the second moment calculates the average of the squared gradients. This approach enhances convergence speed and optimization performance.

In addition, Adam demonstrates strong performance in scenarios involving noisy gradients, such as when training deep learning models with mini batches. It also shows robustness to the selection of hyperparameters.

The update equation of Adam is as follows:

$$\theta_t = \theta_{t-1} - \alpha \frac{\hat{m_t}}{\sqrt{\hat{v_t}} + \epsilon}$$

where θ is the parameters, α_t is the learning rate, $\hat{m_t}$ and $\hat{v_t}$ are estimates of the first and second moments of the gradients, ϵ is the small constant used for numerical stability.

3.4.2. Stochastic Gradient Descent

SGD is an optimization algorithm known for its capability to efficiently handle large datasets. Unlike traditional gradient descent methods that compute gradients using the entire dataset, SGD updates model parameters based on a random sample of data points at each iteration. This characteristic also renders SGD appropriate for scenarios involving continuous generation and addition of new data to the dataset.

Additionally, SGD can navigate non-convex optimization landscapes with multiple local minima by updating parameters with a small number of data points at a time.

The update rule for SGD is as follows:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta L(\theta_t)$$

where θ is the parameters, η is the learning rate, and $\nabla_\theta L$ is the loss function computed for a small subset.

3.5 Measurements

To evaluate the results, we will focus on four performance metrics: accuracy, precision, recall, and F1-score. Accuracy will measure the overall correctness of the model predictions. Precision will evaluate the accuracy of positive predictions, while recall will assess the ability of the models to identify all relevant instances. The F1-score, as the harmonic mean of precision and recall, will provide a balanced measure of performance. These metrics will be calculated for each model on the test set to provide a comprehensive comparison of their performance, helping us in determining the most effective model for our flower classification task.

Additionally, we will include confusion matrix as a measurement tool. A confusion matrix provides a detailed breakdown of the model's predictions, allowing us to examine the true positives, true negatives, false positives, and false negatives to understand the types of errors made by the model.

Furthermore, we will consider prediction time and parameter size of the models. Prediction time measures the time taken by the models to make a prediction on new data, whereas parameter size measures the total number of parameters, both trainable and non-trainable, within the models.

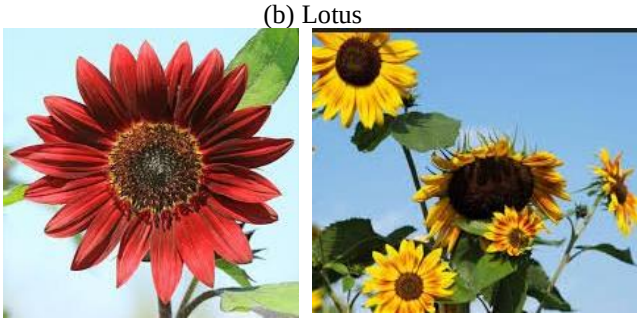
4. DATASET

The dataset used in this project is the 5 Flower Types Classification Dataset, which is publicly available on Kaggle. The dataset contains five distinct flower types: Lily, Lotus, Sunflower, Orchid, and Tulip, with each category comprising 1,000 images.

The images in the dataset are in JPG format with varying resolutions. The images of flowers, even of the same type, differ in lighting, background, colors, and angles, as illustrated in Figure 1.

(a) Lily

(b) Lotus



(c) Sunflower

(d) Orchid

(e) Tulip

Fig. 1. Variations within the same species of flower

5. EXPERIMENT

5.1 Image Preprocessing

Image processing is a necessary step to prepare deep learning models for training and validation in order to identify flower species.

1) *Load Image:* Images from the dataset are extracted from the compressed file, and the extraction process is verified to ensure all images have been successfully loaded for further processing.
2) *Create DataFrame:* Flower image data are organized and labelled by iterating over predefined classes (i.e., Tulip, Lily, Lotus, Sunflower, and Orchid), constructing file paths for each image, and storing these paths with corresponding labels in a list. A DataFrame is then created with columns for file paths and labels, facilitating easy handling of the dataset in subsequent preprocessing steps.
3) *Data Split:* The dataset is split into train, validation, and test sets with proportions of 64%, 16%, and 20%, respectively. The random state is set to 42 to ensure reproducibility of results across different runs.
4) *Resize and Normalize:* The image data is resized to a standard size of 224x224 pixels with three color channels, batched into groups of 32 images per batch, and normalized to a range of [0,1].

5.2 Model

Three different model architectures are implemented for the classification task in this project: DenseNet-121, InceptionV3, and a custom CNN. Transfer learning techniques are applied to DenseNet-121 and InceptionV3, while custom CNN is trained from scratch.

1) *DenseNet-121 and InceptionV3:* Both models utilize pre-trained weights from ImageNet. Each model's output layers are adjusted similarly to include a flattening layer, a fully connected layer with 256 hidden units, a dropout layer with a rate of 0.2, and a final softmax layer tailored for 5-class classification. These modifications ensure compatibility with our specific classification task, maintaining consistency across the architecture adjustments made for both models
2) *Custom CNN:* A custom CNN architecture is designed specifically for this image classification problem. The architecture comprises several convolutional layers with progressively increasing filter sizes. These convolutional layers are interleaved with max pooling layers for downsampling and dropout layers to prevent overfitting. The final layers consist of a flattening layer, a batch normalization layer, a densely connected layer with 256 hidden units, and a softmax layer for classification into five classes.

For training purposes, all models are optimized using both Adam and SGD optimizers. Each model undergoes early stopping, monitored by validation loss, with a patience of 7 epochs to wait before stopping and restoring the best weights. Each model is trained for 20 epochs with a batch size of 32. Categorical cross-entropy serves as the loss function.

For our experiments, we will compare the performance of the deep learning models in flower classification and evaluate the effectiveness of the Adam and SGD optimizers across these architectures.

6. RESULTS

Table I and Table II show the performance metrics as well as the time and size metrics of various image classification models trained with different optimization algorithms.

TABLE I
PERFORMANCE METRICS

Model	Opt.	Acc.	Prec.	Recall	F1
DenseNet-121	Adam	0.95	0.95	0.95	0.95
	SGD	0.84	0.84	0.84	0.84
InceptionV3	Adam	0.94	0.94	0.93	0.93
	SGD	0.94	0.94	0.94	0.93
Custom CNN	Adam	0.68	0.69	0.68	0.68
	SGD	0.57	0.69	0.57	0.55

TABLE II
TIME AND SIZE METRICS

Model	Opt.	Pred. Time (s)	Param. Size (MB)
DenseNet-121	Adam	5.82	75.84
	SGD	6.96	
InceptionV3	Adam	6.88	142.24
	SGD	5.01	
Custom CNN	Adam	4.74	74.74
	SGD	3.72	

Based on Table I, InceptionV3 consistently achieved high performance in accuracy, recall, precision, and F1 score among the image classification models, regardless of the optimization algorithm used. This demonstrates InceptionV3's robustness to different optimizers. In contrast, the performance of DenseNet-121 and custom CNN was affected by the choice of optimizer. Both models performed better when optimized with Adam, with DenseNet-121 surpassing InceptionV3 under this configuration.

According to Table II, InceptionV3 had a significantly larger parameter size, nearly twice that of DenseNet-121 and custom CNN. While custom CNN had the lowest prediction time, its performance was the least effective among the image classification models.

Overall, we can reasonably conclude that DenseNet-121, when optimized with Adam, was the best performing model considering both accuracy and efficiency. The model achieved a high accuracy of 0.95 while maintaining a relatively small parameter size and comparable prediction time. This makes it suitable for scenarios where both performance and resource efficiency are important.

Next, we analyzed the confusion matrices for each deep learning model trained with the Adam and SGD optimizers on the five-class flower classification task, as shown in Figure 2 to Figure 4.

In general, all models demonstrated effectiveness in classifying Sunflower, achieving the highest number of correct classifications in almost all cases. In contrast, all models struggled to classify Lily correctly across all cases.

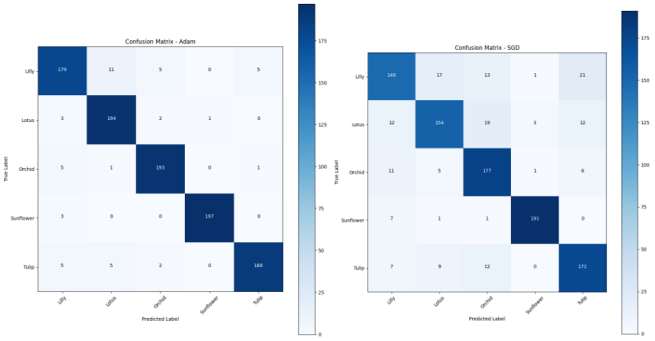


Fig. 2. Confusion matrices of DenseNet-121 with Adam and SGD

DenseNet-121, when optimized with Adam, outperformed SGD, as evidenced by the higher number of correct predictions for all flower types. Notably, SGD encountered difficulty in correctly classifying Lily and Lotus.

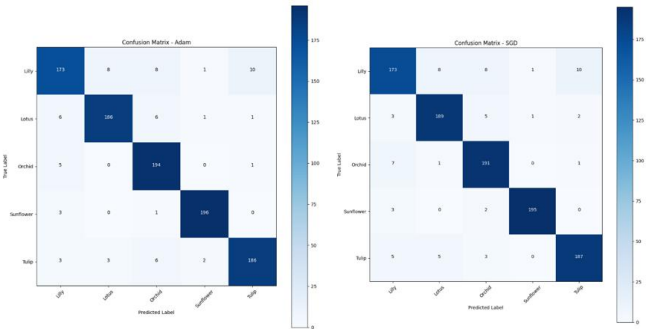


Fig. 3. Confusion matrices of InceptionV3 with Adam and SGD

InceptionV3 performed well with both Adam and SGD, although in both configurations, it occasionally confused Lily with Lotus, Orchid, and Tulip.

Based on Figure 4, the performance of custom CNN with Adam was superior to that with SGD. It is evident that SGD had issues with distinguishing Lily and Orchid, since most of the predictions were wrong for these two classes. Specifically, only 32 instances were correctly classified as Lily, while significant numbers (107) were misclassified as Tulip.

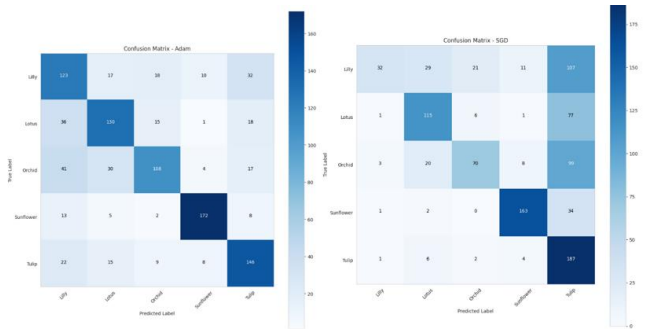


Fig. 4. Confusion matrices of custom CNN with Adam and SGD

To gain deeper insights into these misclassifications, we examined the images using Grad-CAM to visualize the regions of the images that custom CNN with SGD focused on during classification. We started by analyzing the areas used by the model for correctly classifying Lily and Tulip, as shown in Figure 5 and Figure 6.

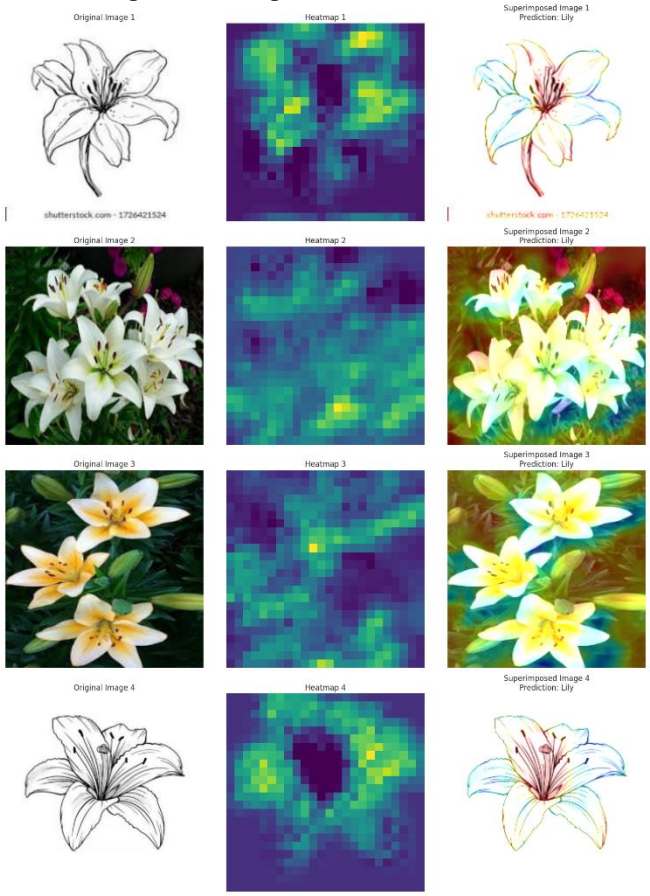


Fig. 5. Grad-CAM visualization for correctly classified Lily images

As illustrated in Figure 5, the model predominantly focused on the petals, suggesting that the petals hold important information that helps the model distinguish Lily from other flower types. Edges and shapes also appeared to be significant for accurate classification of Lily.

According to Figure 6, for the correct classification of Tulip, the model displayed inconsistent focus areas across the images. While all images concentrated on the stems and leaves, indicating that the model considered these regions for classifying Tulip, images 1 and 3 showed additional focus on the petals. This inconsistency could lead to misclassifications, especially when there are similarities in these regions between different flower types.

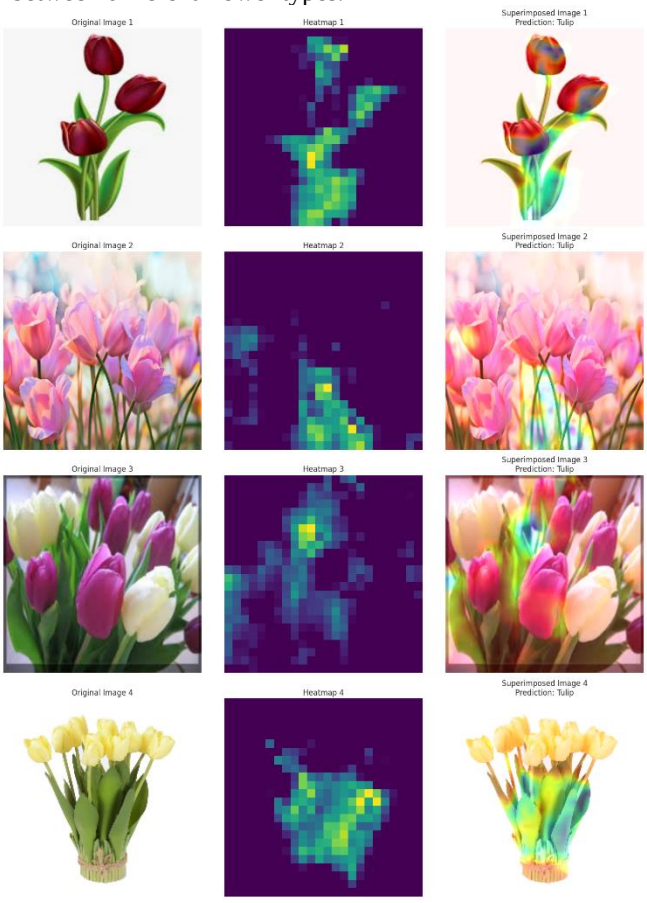


Fig. 6. Grad-CAM visualization for correctly classified Tulip images

Subsequently, we examined the images where Lily was wrongly classified as Tulip to understand the model's confusion.

Based on Figure 7, the model failed to capture the complete petals of Lily, leading to incomplete feature extraction and misclassification. In addition, in some of the misclassified images, Lily petals resemble those of Tulip. Therefore, the model often confused Lily for Tulip.

Similarly, only 70 instances were correctly classified as Orchid, with a substantial number (99) being misclassified as Tulip.

Moreover, while 115 instances were correctly identified as Lotus, the number of misclassifications was still notable, particularly towards the Tulip class (77 instances).

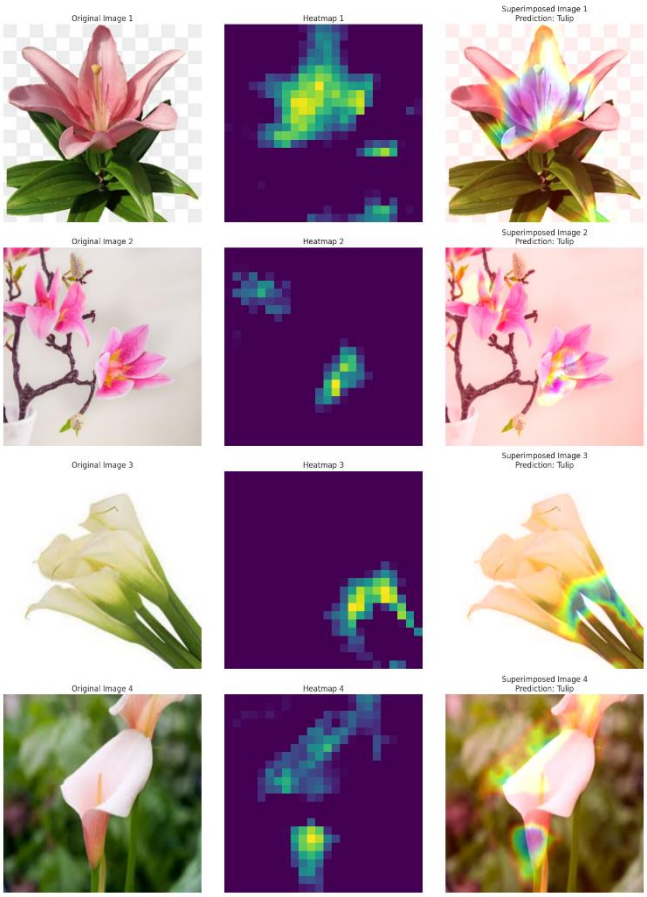


Fig. 7. Grad-CAM visualization for Lily images misclassified as Tulip

8. CONCLUSION

This project aimed to enhance flower classification using deep learning methods, such as DenseNet-121, InceptionV3, and a custom CNN, optimized with Adam and SGD.

Based on our experiment, DenseNet-121 with Adam emerged as the best performing model with an accuracy of 95%, achieving high performance without compromising computational efficiency. Notably, Adam outperformed SGD when used as an optimization algorithm with DenseNet-121 and custom CNN. Meanwhile, InceptionV3 demonstrated robustness by achieving high performance across different optimizers.

Our analysis revealed challenges in classifying Lily, particularly with the custom CNN optimized with SGD. This model recorded a significant number of misclassifications, often confusing Lily with Tulip due to inconsistent focus areas and similar petal shapes.

Future improvements include using advanced architectures capable of capturing finer details and improving feature differentiation, applying sophisticated data augmentation techniques, utilizing more diverse datasets to enhance model performance and generalizability. Specifically, it is crucial to incorporate a wider variety of training images of Lily and Tulip, focusing on challenging cases where their features overlap.

9. REFERENCES

[1] S Kishotha, & B. Mayurathan. (2019). Machine Learning Approach to Improve Flower Classification Using Multiple Feature Set. https://doi.org/10.1109/iciis47346.2019.9063349

[2] Mete, B. R., & Ensari, T. (2019). Flower Classification with Deep CNN and Machine Learning Algorithms. 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT). https://doi.org/10.1109/ismsit.2019.8932908

[3] Qin, M., Xi, Y., & Jiang, F. (2019, December 1). A New Improved Convolutional Neural Network Flower Image Recognition Model. IEEE Xplore. https://doi.org/10.1109/SSCI44817.2019.9003016

[4] He, M., Zhu, H., Li, Y., Tang, S., & Sun, Z. (2021). Flower Image Classification Based on Multi-scale Dense Residual Network. https://doi.org/10.1109/icivc52351.2021.9526970

[5] Pawar, S., Raj, W., Suruchi Bibikar, Patil, B., Kumari, S., & Patil, A. (2022). Identification of Flowers using Machine Learning. 2022 International Conference on Innovations in Science and Technology for Sustainable Development (ICISTSD). https://doi.org/10.1109/icistsd55159.2022.10010418

[6] Sanad. (2024, June 6). Image Classification Using CNN (Convolutional Neural Networks). Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/

[7] Craig, L. (n.d.). Convolutional Neural Network (CNN) (R. Awati, Ed.). TechTarget. https://www.techtarget.com/searchenterpriseai/definition/convolutional-neural-network

[8] Ruiz, P. (2018, October 11). Understanding and visualizing DenseNets. Medium; Towards Data Science. https://towardsdatascience.com/understanding-and-visualizing-densenets-7f688092391a

[9] Raj, B. (2018, May 30). A Simple Guide to the Versions of the Inception Network. Medium; Towards Data Science. https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202

[10] Jamhuri, M. (2023, April 15). Understanding the Adam optimization algorithm: A deep dive into the formulas. Medium. https://jamhuri.medium.com/understanding-the-adam-optimization-algorithm-a-deep-dive-into-the-formulas-3ac5fc5b7cd3

[11] Mishra, M. (2023, June 5). Stochastic Gradient Descent: A basic explanation. Medium. https://mohitmishra786687.medium.com/stochastic-gradient-descent-a-basic-explanation-cbddc63f08e0

[12] Arun. (2024, July 4). Unconstrained optimization techniques in Neural Networks. GeeksforGeeks. https://www.geeksforgeeks.org/unconstrained-optimization-techniques-in-neural-networks-1/

[13] 5 Flower Types Classification Dataset. (n.d.). Www.kaggle.com. https://www.kaggle.com/datasets/kausthubkannan/5-flower-types-classification-dataset/