# Flower Identification

Huang Yujie
*Faculty of Computing, Software Engineering, Cyberspace Security*
*Nanjing University of Posts and Telecommunications*
Jiangsu, Nanjing, China
847993517@qq.com

***Abstract*—The objective of this paper is to evaluate and compare the performance of different convolutional neural network (CNN) models in 24 types of flower image recognition tasks through a series of experiments in order to identify the best models. The main algorithms studied include ResNet50, VGG19, and InceptionV3, which have good performance in the field of deep learning and computer vision and are widely used. By training and testing these models on the same dataset and under the same experimental conditions, this paper compares their various evaluation indicators. The experimental results show that ResNet50 performs better in the flower image recognition task. ResNet50 is known for its deep residual learning framework, which can effectively solve the problem of vanishing gradients in deep networks, enabling the network to improve performance by learning residual mappings. In contrast, VGG19, while excellent at image recognition tasks, results in a slower training and inference process due to its deep network structure and large number of parameters. InceptionV3, for its part, uses the Inception module to enhance the depth and width of the network, and while it performed well in some tasks, it did not outperform ResNet50 in the 24-category flower image recognition.**

***Index Terms*—Flower recognition, CNN, ResNet50, VGG19, InceptionV3, image classification**

## I. Introduction

Flower identification technology has been widely used in many fields in recent years, especially in agriculture, horticulture, ecological monitoring and intelligent mobile applications. With the development of computer vision and artificial intelligence technology, especially the breakthrough of convolutional neural network (CNN) in the field of image processing, the precision and efficiency of flower recognition have been significantly improved. Through multi-layer convolutional, pooling and fully connected layer structure, CNN can automatically learn the deep feature representation from a large number of flower images, so as to effectively distinguish different kinds of flowers. Compared with the traditional methods based on artificial feature extraction, CNN does not need to rely on complex artificial design features, and can automatically extract more recognizable features, especially in the processing of complex background, different shooting angles and lighting conditions of flower images. This makes the application of flower recognition technology in agricultural planting, pest monitoring, plant protection and other fields more accurate and efficient. In addition, flower identification technology also plays an important role in environmental protection and ecological monitoring, helping researchers assess changes in the ecological environment and understand the distribution of plant communities through automatic identification and monitoring of plant species in large-scale areas.

## II. Data collection and pre-processing

### A. Data sets

The data set for this experiment was obtained from online sharing and included 48,000 photos of 24 species of flower plants. The flowers were one year, clover, Bougainia, two colors of Cinquea, Whole leaf of Mala, whole border of golden light, sword leaf of Cinquea, Milion, Motherina, Archery, convolvula, Datura, Platycodon, rapeseed, shore chrysanthemum, setaria, wolftail, dragon ray, hydrangea coronet, dandelion, blue thistle, jackweed, verbena and bidenwort.

### B. Image clean-up

Noise removal: The image may contain noise that interferes with the learning process of the model. The noise is removed by an image denoising algorithm.

### C. Image size processing

To ensure that the input data fed into the model has a consistent shape, all flower images are uniformly resized. The default input resolution is set to $224 \times 224$ pixels, which aligns with the standard input size required by most convolutional neural networks such as VGG19, ResNet50, and InceptionV3.

### D. Image normalization and standardization

Pixel value normalization: Typically the pixel value of an image ranges from 0 to 255. When training a neural network, you need to normalize the pixel values to between 0 and 1, or use normalization to convert the pixel values to a distribution with a mean of 0 and a variance of 1. We divide the pixel values by 255 and normalize them with the mean and standard deviation for each channel.

### E. Data Enhancement

Usually when the data set is small, it will increase the diversity of the training data by randomly rotating the image or flipping the image horizontally and vertically to avoid overfitting the model to a specific direction or position. Or randomly crop or pan the image to simulate the situation of flowers in different viewing angles and sizes. Due to the large data set, this paper has been able to train the model to a better effect, so there is no step of data enhancement.

## III. Related technical Research

In recent years, the application of convolutional neural networks in flower recognition has received extensive attention and in-depth research. As a typical image classification problem, flower recognition is faced with a lot of challenges, including a wide variety of flowers, complex image background, lighting changes, similarities in flower morphology and inconsistent shooting angles. Traditional image processing and machine learning methods tend to show low recognition accuracy and robustness when dealing with these problems, while CNN has shown strong advantages in this respect through its unique feature extraction mechanism. CNNS are able to automatically learn hierarchical features in images through multiple convolutional layers, thus achieving efficient and accurate classification. With the continuous development of CNN architecture, deep network models such as VGG, Inception and ResNet have been widely applied in flower recognition tasks, and remarkable results have been achieved.

### A. *ResNet*

ResNet, or residual network [1], is a deep learning model proposed by Microsoft Research to solve the problem of disappearing gradients and exploding gradients in deep neural network training. The core idea of ResNet is to introduce the residual learning framework, by adding cross-layer connections, the input is directly added to the output, allowing the gradient to pass directly through these connections, so that the network can effectively train deeper hierarchies. The formula is as follows:

$$y = \mathcal{F}(x, \{W_i\}) + x \tag{1}$$

Where $\mathcal{F}(x, \{W_i\})$ is a map that extracts features through several convolution layers, where $x$ is the input and $y$ is the output. With this form, the network learns not only $\mathcal{F}(x)$, but the residuals from the input $x$ ($\mathcal{F}(x)+x$). Each Residual Block usually consists of two or three convolutional layers, and the input is added directly to the output of each block.

With this innovation, ResNet was able to train deeper and more complex networks, and it achieved remarkable results in flower recognition [2]. Especially in the case of a large number of flower species and complex images, ResNet is able to extract more detailed and abstract features through the deep network, thus improving the accuracy of classification. Because the deep network can learn more rich feature representations, ResNet is particularly suitable for fine-grained classification in flower images, and can effectively distinguish flower species with similar morphology.

ResNet models of different depths mainly include ResNet-18, ResNet-34, ResNet-50, ResNet-101 and ResNet-152 [3]. Considering the experimental training time and other factors, the ResNet-50 model is chosen in this paper.

ResNet50 is composed of multiple convolutional layers, batch normalization layers and activation function layers, including a total of 50 convolutional layers. The network is divided into five stages. Stage0 is relatively simple and can be regarded as data preprocessing. The latter stages, Stage1, Stage2, Stage3, and Stage4, all Bottleneck structures are similar.

Stage0 stage (3, 224, 224) refers to the input channel number (channel), height (height) and width (C, H, W). Now assume that the input height and width are equal, that is, the shape is (C, W, W). In this stage, the input data will pass the convolution layer, BN layer, ReLU activation function and MaxPooling layer to get the output shape of (64, 56, 56).

Among them, the convolution layer includes a $7 \times 7$ convolution kernel with stride 2 and output 64 channels; The max pooling layer has a kernel size of $3 \times 3$ and a stride of 2. The BN layer, also known as Batch Normalization, can alleviate the internal covariate deviation, make the input distribution of each layer more stable, thus speeding up the convergence, helping to speed up the training process, stabilize the training, and improve the final performance of the model.

Residuals in ResNet50 use a Bottleneck structure, that is, the bottleneck residuals (BTNK in the figure) that occur in Stage1, Stage2, Stage3, and Stage4 are divided into two situations: Convolutional Block is used to change the dimension of the network (BTNK1 in the figure), while Identity Block is used to deepen the network (BTNK2 in the figure). The specific structure is shown in the Fig. 1.

An **Identity Block** is a residual block in which the input and output dimensions are identical. That is, the skip connection remains unchanged and the input is added directly to the output without any dimensional transformation. The block includes three convolutional layers (along with their corresponding Batch Normalization (BN) layers and ReLU activation functions):

- **1×1 convolution**: Reduces the number of channels in the input feature map, typically compressing the channel size to form a bottleneck layer.
- **3×3 convolution**: Performs spatial feature extraction on the reduced-channel representation.
- **1×1 convolution**: Restores the number of channels back to the original dimension.

Let the input $x$ be of shape $(C, W, W)$, and let the composite of the three convolutional layers be denoted as function $\mathcal{F}(x)$. The residual output is then computed as:

$$y = \mathrm{ReLU}(\mathcal{F}(x) + x) \tag{2}$$

The output shape remains the same as the input: $(C, W, W)$.

**Convolutional Blocks** are structurally similar to Identity Blocks but differ in that they allow the input and output dimensions to vary. This is necessary when either the spatial dimensions (height and width) or the number of channels changes between layers.

To match dimensions, Convolutional Blocks include an additional convolutional layer on the shortcut path (right side), which transforms the input into the same shape as the output. Let this transformation be denoted as $\mathcal{G}(x)$. Then the residual computation becomes:

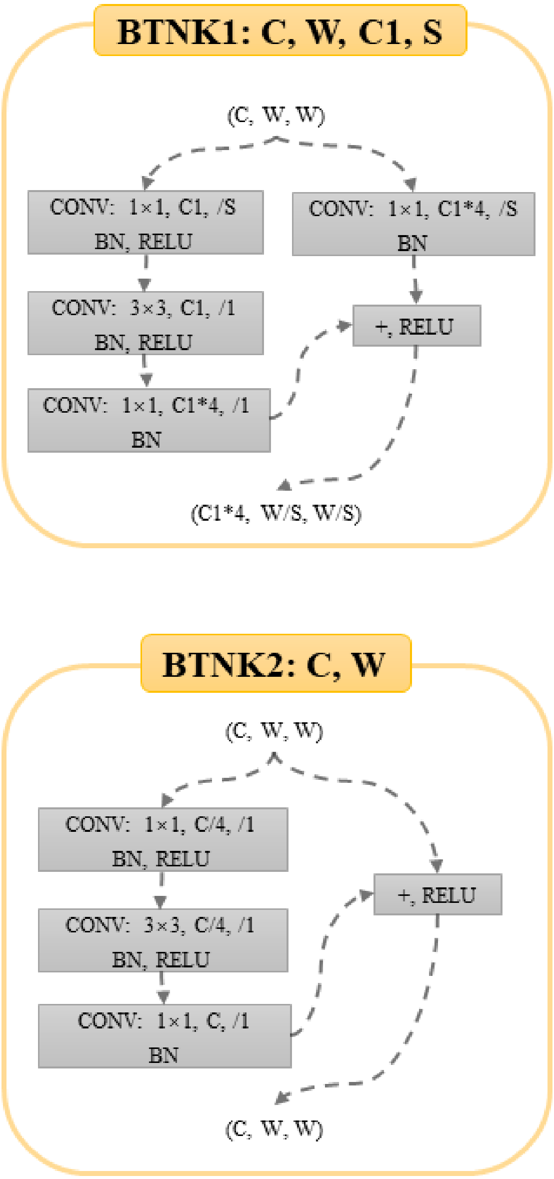

Fig. 1. Structure diagram of the bottleneck residual block.

$$y = \mathrm{ReLU}(\mathcal{F}(x) + \mathcal{G}(x)) \tag{3}$$

The parameters of Identity Blocks in Stage 1, Stage 2, Stage 3, and Stage 4 are largely consistent, while Convolutional Blocks vary as follows:

- **Stride** $S$: Indicates whether subsampling is performed in the two $1 \times 1$ convolutions.
  - In the Convolutional Block of **Stage 1**, $S = 1$, no subsampling is applied, and the input and output dimensions remain equal.
  - In the Convolutional Blocks of **Stages 2–4**, $S = 2$, subsampling is applied, and the input spatial resolution is twice that of the output.
- **Channels** $C$ **and** $C_1$: Refer to the number of input channels and the number of output channels in the first $1 \times 1$ convolution on the left path.
  - In **Stage 1**, $C = C_1 = 64$, meaning the first $1 \times 1$ convolution does not reduce channel count.
  - In **Stages 2–4**, $C = 2 \times C_1$, so the first $1 \times 1$ convolution reduces the number of channels by half.

### B. *Vgg*

VGG19 is a deep convolutional neural network proposed by the Visual Geometry Group at the University of Oxford [4]. It is a variant in the VGG series consisting of 19 weighted layers, including 16 convolutional layers and 3 fully connected layers. The design of the VGG network is simple and effective, characterized by a concise and repeatable structure.

Each convolutional layer in VGG19 uses a $3 \times 3$ kernel, and each pooling layer uses a $2 \times 2$ kernel. This consistent design makes the model straightforward and easy to implement. The core feature of the **VGG** network is its strategy of extracting image features through stacking multiple convolutional layers with small-size kernels. This architecture enables the network to capture local features effectively and gradually learn more abstract high-level features by increasing depth.

In the context of flower recognition, VGG19 can utilize its deep architecture to extract fine-grained visual features, particularly capturing subtle morphological variations in petals, leaves, and buds. However, VGG19 also has drawbacks. Due to the large number of fully connected layers—especially the last one—it contains a significant number of parameters. This results in high computational cost during training and inference, making it less efficient compared to more optimized architectures.

### C. *Inception*

Inception [5] is a convolutional neural network architecture whose key design concept is to combine convolutions of different kernel sizes in parallel. It enhances the representational capacity of the network through **multi-scale feature extraction**, aiming to achieve a better balance between classification accuracy and computational efficiency. Due to these characteristics, Inception is widely used in tasks such as image classification and object detection.

The model incorporates convolution kernels of various sizes (e.g., $1 \times 1$, $3 \times 3$, $5 \times 5$) and pooling operations (e.g., $3 \times 3$ max pooling) within a single module. These operations are executed in parallel, and their resulting feature maps are concatenated to form a multi-channel output. This enables the network to extract and integrate information across multiple scales efficiently.

In addition, the Inception network introduces **auxiliary classifiers**, which are added to intermediate layers of the network. These classifiers support local training and help improve gradient flow, thereby alleviating the vanishing gradient problem. This feature makes the architecture suitable for training on large-scale datasets and for handling complex tasks.

Unlike the VGG network, Inception adopts a “**breadth-first**” design strategy, allowing each layer to use different convolution and pooling operations. This provides the network with diverse receptive fields at each level of processing, enhancing its ability to extract complex features.

Such a design is particularly well-suited for flower image recognition, where different parts of the flower (e.g., petals, calyx, leaves) may exhibit distinct features at varying scales. Inception’s ability to process these heterogeneous visual patterns simultaneously contributes to improved recognition accuracy.

## IV. Flower recognition

### A. Experimental setup

The dataset for this experiment included 24 species of flower plants with a total of 48,000 photos. The flowers were one year, clover, Bougainberry, two colors of Cinquea, whole leaf of Mala, whole border of golden light, sword leaf of Cinquea, Milion, motherina, shoot stem, convolvula, Datura, Platycodon, rapeseed, shore chrysanthemum, setaria, wolftail, dragon ray, hydrangea coronet, dandelion, blue thistle, jackweed, verbena, and bidenwort. Training set: Validation set: test set 8:1:1.

### B. Main code

1) **Identity Block:** This block is a core module in the ResNet architecture. The parameter `filters` is disassembled into three values, denoted as $F_1$, $F_2$, and $F_3$, representing the number of output channels for the three convolutional layers. These values are then used to define the channel dimensions of each convolution.

   The original input $X$ is stored as a shortcut connection, denoted as $X_{\text{shortcut}}$, and preserved for residual addition. After the main path completes the three-layer convolution, the output is added to $X_{\text{shortcut}}$ to implement residual learning:

$$\text{Output} = \text{ReLU}(\mathcal{F}(X) + X_{\text{shortcut}}) \tag{4}$$

   This residual connection helps mitigate the vanishing gradient problem by enabling the network to learn identity mappings more easily. Finally, a ReLU activation function is applied to ensure nonlinearity.

2) **Convolutional Block:** This block is also a core component in constructing deep residual networks. The main path consists of three convolutional layers, each followed by a Batch Normalization (BN) layer and a ReLU activation function.

   First, a $1 \times 1$ convolution is applied to the input $X$ with stride $s$, using `glorot_uniform` as the kernel initializer. The feature map size changes according to the stride. After convolution, the result is normalized using BN and then passed through ReLU.

   The second convolution uses an $f \times f$ kernel with stride 1 and `padding='same'` to maintain the spatial dimensions. The third convolution is another $1 \times 1$ convolution with stride 1.

   To match the output dimensions of the main path, the shortcut path $X_{\text{shortcut}}$ is adjusted using a $1 \times 1$ convolution with $F_3$ output channels and stride $s$, followed by BN.

   Finally, the outputs of the main path and the shortcut path are added element-wise:

$$\text{Output} = \text{ReLU}(\mathcal{F}(X) + \mathcal{G}(X_{\text{shortcut}})) \tag{5}$$

   This element-wise addition is the core of residual connection. The result after summation is passed through a ReLU activation to introduce nonlinearity and maintain the learning capacity of the deep network.

## V. Test results and evaluation

The evaluation index changes of each iteration of ResNet50 are shown in Figs. 2–6

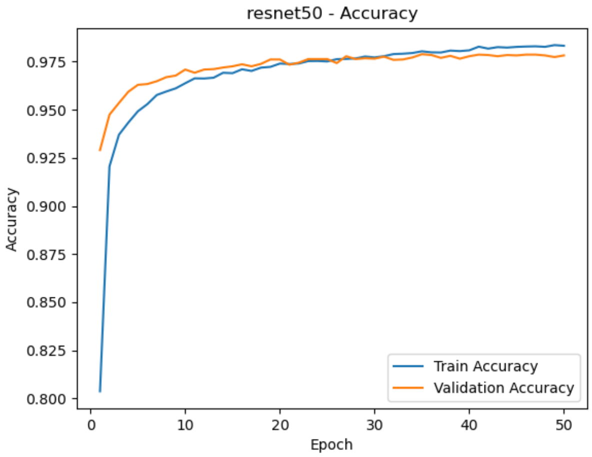

Fig. 2. Accuracy of ResNet50.

After 30 rounds of Epoch, each index tends to converge and there is no significant change anymore, and the model has approached its optimal performance.

TABLE I
Comparison of Evaluation Indexes of Each Model Test Set

| Model | Accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|
| ResNet50 | 0.979480 | 0.979490 | 0.979528 | 0.979443 |
| VGG19 | 0.943257 | 0.944067 | 0.943228 | 0.943302 |
| Inception_v3 | 0.932788 | 0.933122 | 0.932917 | 0.932780 |

As shown in Table I, there is a comparison of the evaluation indicators for each model’s test set. As shown in Table I, ResNet50 achieves the best performance in the flower recognition task, with higher accuracy, precision, recall, and F1-score values than both VGG19 and Inception_v3.

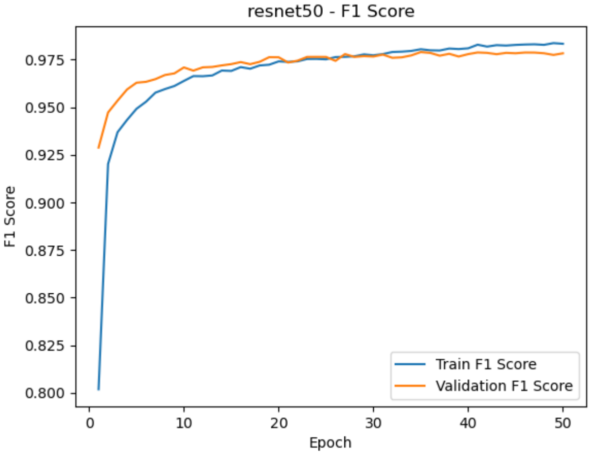

Fig. 3. F1 Score of ResNet50.

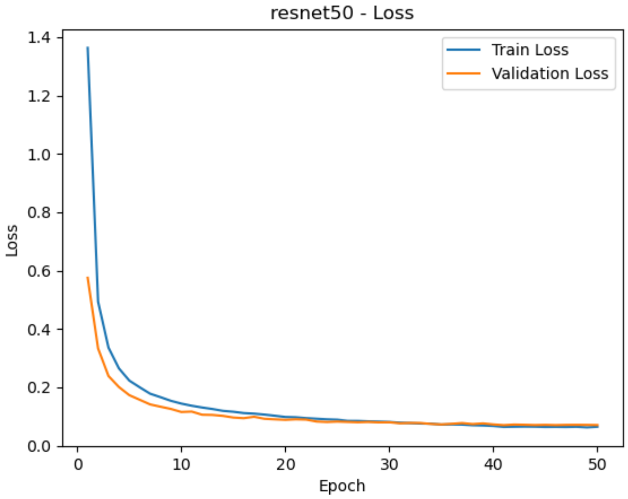

Fig. 4. Loss of ResNet50.

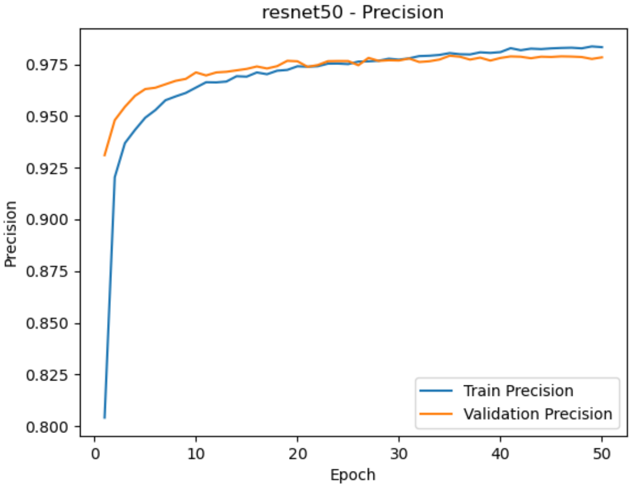

Fig. 5. Precision of ResNet50.

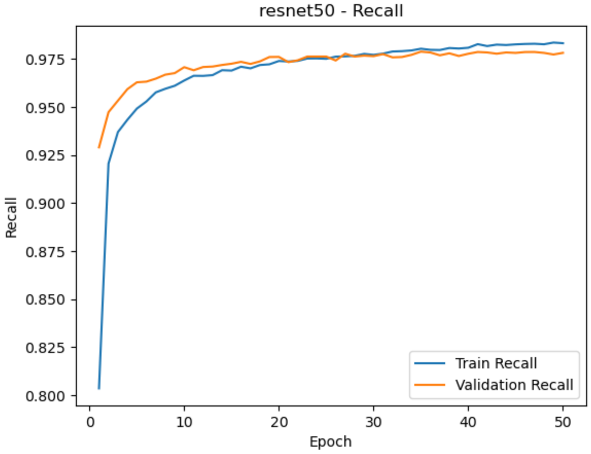

Fig. 6. Recall of ResNet50.

Specifically, ResNet50 achieves an accuracy of 97.95%, precision of 97.95%, recall of 97.95%, and an F1-score of 97.94%. These results indicate that ResNet50 not only maintains high precision in the recognition process, but also effectively identifies the majority of positive samples.

In comparison, VGG19 achieves an accuracy of 94.33%, precision of 94.41%, recall of 94.32%, and an F1-score of 94.33%. Although slightly less performant than ResNet50, it still shows reliable classification capability.

Inception_v3 shows the lowest performance among the three, with accuracy, precision, recall, and F1-score values of 93.28%, 93.31%, 93.29%, and 93.28% respectively.

Overall, due to its excellent comprehensive performance across all evaluation metrics, ResNet50 is the most recommended model for the flower recognition task, while VGG19 and Inception_v3, although close in performance, are slightly inferior.

## CONCLUSION

The superior performance of ResNet50 in the flower recognition task may be closely attributed to its deep residual learning framework. By utilizing residual connections, ResNet50 effectively mitigates the problems of gradient vanishing and overfitting in deep neural networks, thereby enhancing the overall training effectiveness.

In the image classification task involving 24 types of flowers, ResNet50 is capable of learning more abstract and high-level features through its residual blocks. The deep network architecture enables it to extract finer image details, leading to improved recognition accuracy. This efficient learning mechanism allows ResNet50 to reliably recognize floral features even in complex and diverse flower images, reducing the risk of misclassification.

Although VGG19 and Inception_v3 also perform well, they are slightly inferior to ResNet50. The potential reason is that, compared to VGG19, ResNet50 incorporates more convolutional filters with smaller kernel sizes, which improves both computational efficiency and parameter management. While VGG19 has a deep architecture, it comes with a significantly larger number of parameters and higher computational complexity. On the other hand, Inception_v3 improves efficiency via its modular structure but is architecturally more complex and may not be optimized for fine-grained classification tasks, such as flower recognition with minimal inter-class variation.

Experimental results consistently demonstrate that among the three models—ResNet50, VGG19 and InceptionV3, ResNet50 achieves the best performance on flower recognition tasks. This is primarily due to its deep residual learning mechanism, which enables it to learn efficiently on complex datasets. Furthermore, ResNet50 exhibits strong generalization capability, making it both accurate and highly adaptable.

## References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, 2016, pp. 770–778.

[2] S. Jaju and M. Chandak, "A Transfer Learning Model Based on ResNet-50 for Flower Detection," in *Proc. Int. Conf. Appl. Artif. Intell. Comput. (ICAAIC)*, Salem, India, 2022, pp. 307–311.

[3] P. Sarikabuta and S. Supratid, "Impacts of Layer Sizes in Deep Residual-Learning Convolutional Neural Network on Flower Image Classification with Different Class Sizes," in *Proc. Int. Elect. Eng. Congr. (iEECON)*, Khon Kaen, Thailand, 2022, pp. 1-4.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[5] X. Xia, X. Cui, and B. Nan, "Inception-v3 for flower classification," in *Proc. Int. Conf. Image, Vision and Comput. (ICIVC)*, Chengdu, China, 2017, pp. 783–787.