NatureLens: Smart plant detection to assist customers and optimize inventory

Euraika M. Borja  
Bulacan State UniversityCity of Malolos, Bulacan  
2021100715@ms.bulsu.edu.ph

Genesis Nathan V. Torres  
Bulacan State UniversityCity of Malolos, Bulacan  
2021104898@ms.bulsu.edu.phGhyn Harry B. Cabote  
Bulacan State UniversityCity of Malolos, Bulacan  
2021106753@ms.bulsu.edu.ph

Paul Andrei M. Pangilinan  
Bulacan State UniversityCity of Malolos, Bulacan  
2021103775@ms.bulsu.edu.ph

***Abstract*—This paper introduces NatureLens, a smart detection app that simplifies plant identification and floral inventory management. Using advanced scanning and image recognition, NatureLens helps people easily identify plants and access care information. It also enables businesses to optimize inventory by tracking stock levels, trends, and customer preferences in real-time. By incorporating AI and machine learning, NatureLens aims to enhance the peoples experience and improve inventory management for both consumers and florists.**

# **Introduction**

The application NatureLens is useful to both florists and consumers by mitigating multiple chores involved in such business. To the florist, it means that they do not need to regularly check what flowers they currently have in their store or catalogue and what the customers are most likely to buy in order to avoid stocking unnecessary flowers that will go to waste. For the consumer, the application helps with identifying plants and consultation on the choice as well as care in choosing plants and their care. Most especially to gardeners and homeowners, it is very useful. Moreover, NatureLens also helps the user in identification of the plants in unfamiliar locations which takes a lot of time because the app as soon as provides the details of the plant on our screen. Pains in creating the app are the ability of the resulting app to accurately recognize plants, the timeliness of the information processing, compatibility with any inventory system, and maintainability and affordability of the process.

# **Related Literature**

**The future of gardening: exploring the potential of smart plant sensors** - The growing use of smart plant sensors in gardening is changing the face of conventional gardening and improving plant welfare. As exemplified by recent developments, such devices present a harmony of orthodox horticulture comparable to the experience provided by an experienced horticulturist and new age innovations which include tools that help monitor important environmental conditions including moisture, temperature of the soil, light intensity, and even nutrient content within the soil. Those sensors, with the help of artificial intelligence, allow gardeners to make correct guesses about the plant watering, fertilizer application, plant position, etc.. Self-watering pots contain sensor technology that allows them to provide water for plants automatically, which alongside other improvements are making gardening easier than ever, especially for indoor plants and people with little time on their hands. This innovative process is not only beneficial for the improved plant health and greener approaches to gardening but also for the development of a direct, comfortable relationship between human and plant – opening the age of traditional, manual gardening and high-level technology.

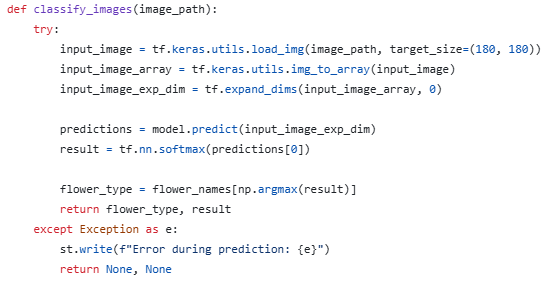
**How a Plant Nursery Management System Can Streamline Inventory Control? -** The technological developments in managing plant nursery inventories are the main factors that have shifted the nurseries from being simple bases to complex ones. Nurseries which used to use traditional methods of record keeping in their business now use plant nursery management software through which the tracking of species, species database and even sales can be done in real time. Sensors for monitoring the internal environment of the plant, AI, and machine learning improve decision-making by increasing order accuracy and forecasting possible illnesses of plants. Some enhancements include IoT, RFID and real-time monitoring are implemented in FarmERP, one of the leading software that revolutionized nurseries, increasing profitability alongside health improvements of plants acquired and better practices.

**Plant Species Identification Using Computer Vision Techniques -** Up to the recent past, plant species identification was a process that demands professional botanical knowledge, and thus was not easily feasible by ordinary citizens and indeed many professionals. This situation has been made worse by the scarcity of skilled taxonomists, therefore exacerbating what is referred to as the “taxonomic crisis.” Nevertheless, with the advent of technology such as carrying portable devices that can identify plants’ through applications and ImageCLEF which is an automated plant identification system, professional training is not a requisite in plant identification. Such technologies like augmented reality and 3D scanning also supplement the case for the use of automatic plant identification technologies for the long-term.

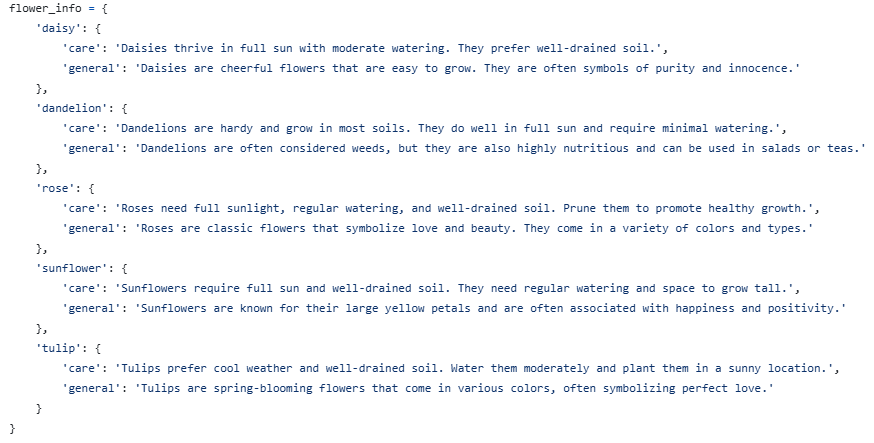
# **Methodology**

This research employs a convolutional neural network (CNN) model to classify five flower species: daisy, dandelion, rose, sunflower and tulip. There were 4317 images in the dataset, with 80% of the data used for training and the rest used for validation; 3454 training samples and 863 validation samples. To enhance the generality of the model horizontal flipping, rotation, and zoom were applied using data augmentation.

CNN model in details: The proposed CNN in this study was implemented using TensorFlow Keras API and consists of convolutional layers, max pooling layers and dropout layers. A flatten layer converts feature maps into one-dimensional vector then dense layers and an output layer with five neurons for classification.  
  
The classify\_images function analyses an image and determines the type of flower with its probability ratings through a model. This is meant to be able to capture mishaps that may occur and provide for friendly usage of the execution results.



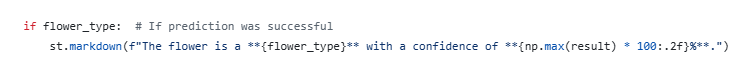
This dictionary provides care and general information about various types of flowers which makes it a data structure.



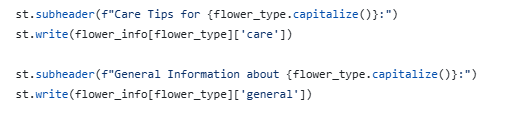
This function takes the image, analyses it, uses a machine learning model to make an approximation as to what kind of image it is, and spits out the result of the approximation. After this output, the code line at hand assigns another section to the flower type, and the result obtained is provided here.



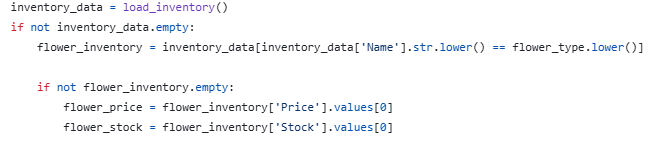
This line of code is used in the feedback system of a Streamlit application. The last one is mainly used to verify a correct prediction and show it with a certain accuracy in the desired format.



This code’s objective is to show care tips more generally about the described flower type using the Streamlit application. It employs st.subheader and st.write to organize and display information most conveniently for a user.

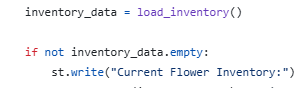


It is this block of code designed to load data about the inventory and using the result of the identification of the flower type, read its price and quantity in stock, according to the data in the dataset.

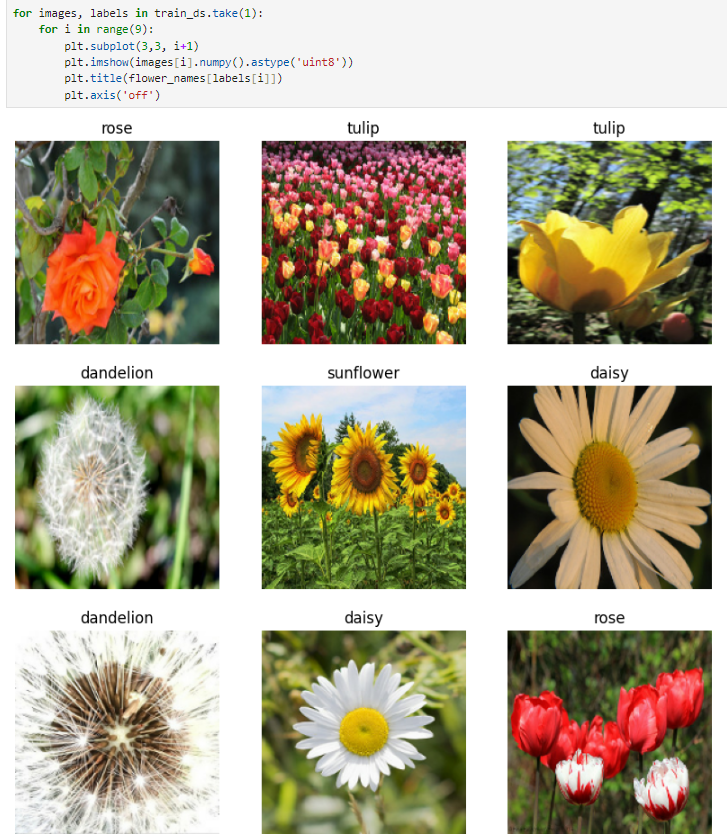


The purpose of this chunk of code is to read inventory data and determine whether or not the inventory is not empty

inventory\_data = load\_inventory()

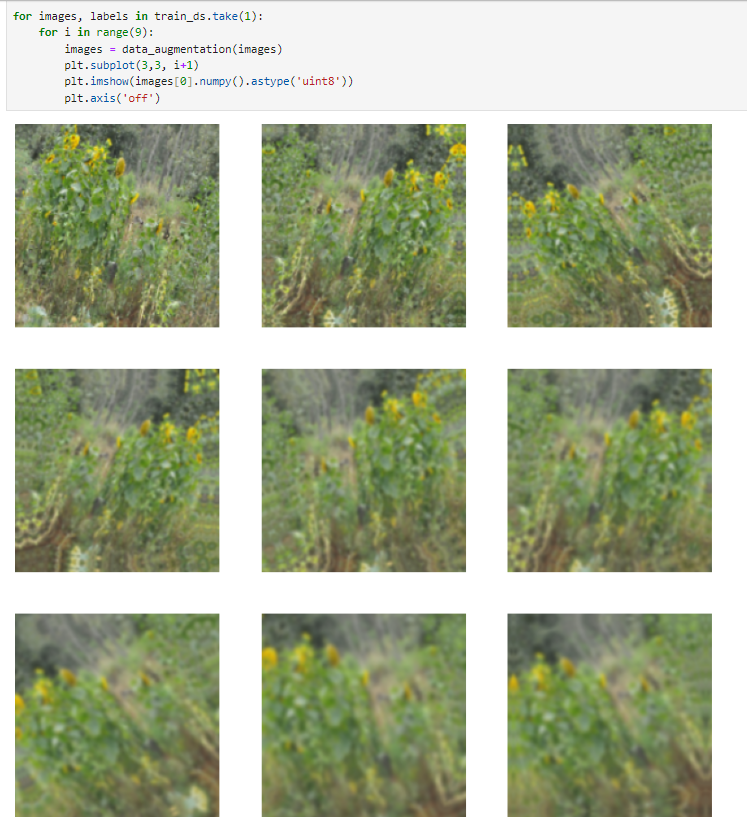


In order for the application to identify the images of the flowers, the developers trained a model using TensorFlow. The developers used the help of the internet to gather data sets to train the model. The data sets are labeled and sorted so that the model can gather as much information about the different looks of the flowers. The model is trained using 5 different flowers, Daisy, Dandelion, Rose, Sunflower and Tulip.

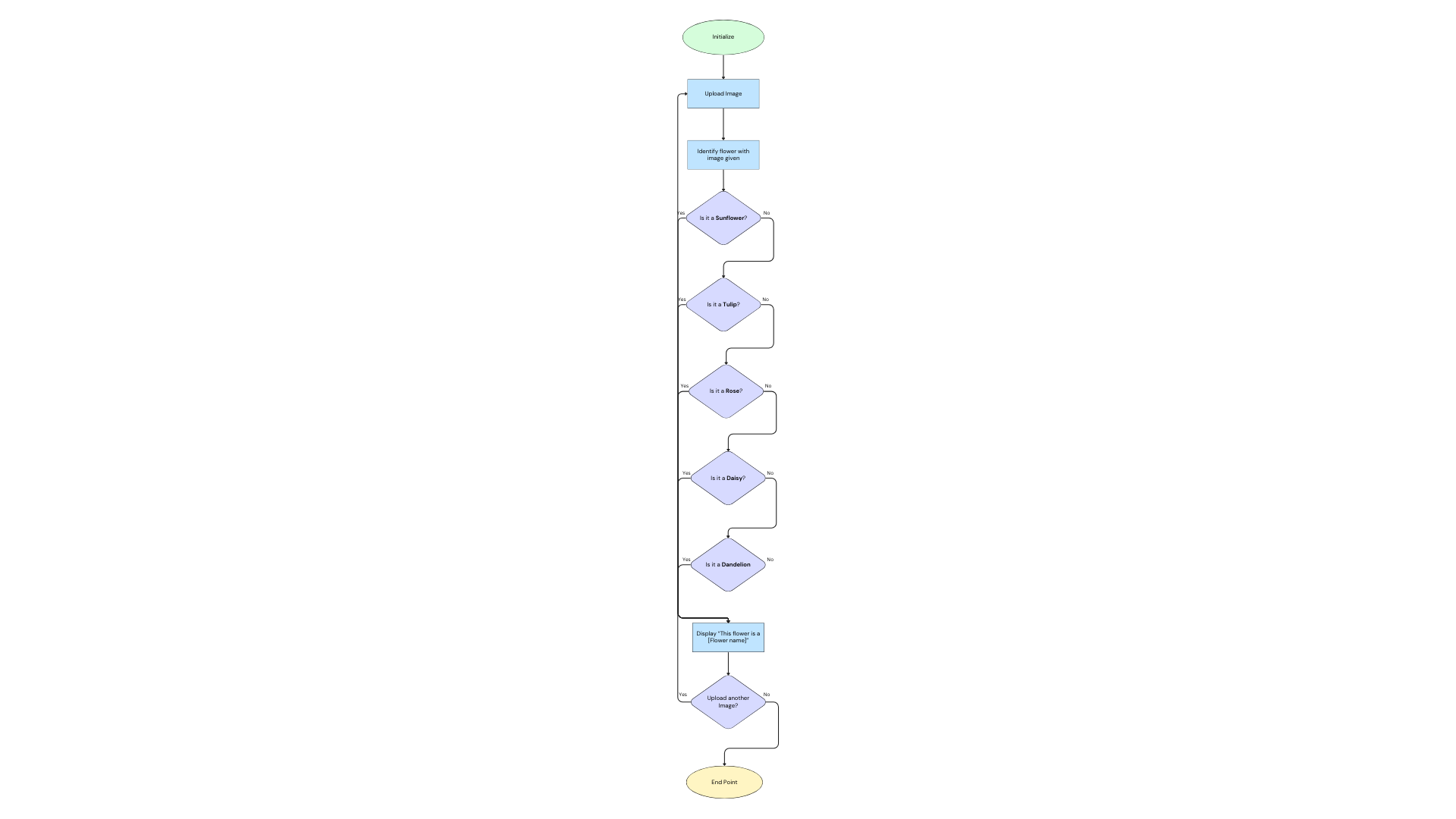


Although data sets are collected, it is not sufficient. We then

used data augmentation to fill up the needed data sets to train the model. Data augmentation is making a new data out of old data, it makes small changes such as, rotating, zooming or changing colors of the old data. Below are the example of data augmentation:

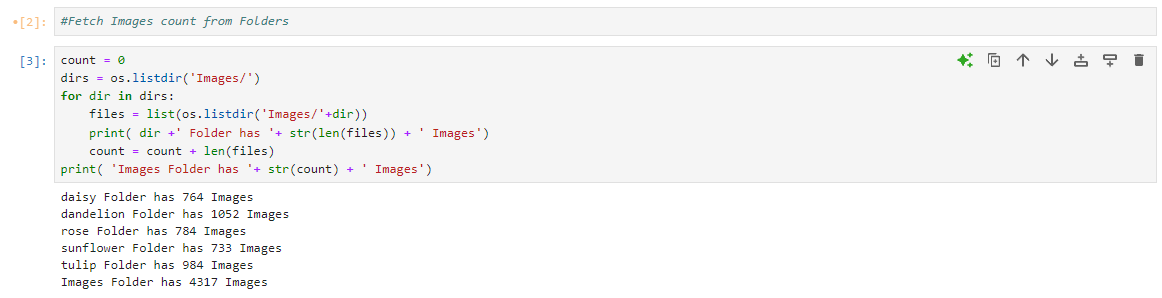
**Flowchart**

This flowchart shows the way to determine types of flowers from the image that has been uploaded. It begins with the loading and determination of an image that indicates the particular type of flower. The flow checks the flower first if it is a Sunflower then tests if it is a Tulip and so on until it is either a Daisy or Dandelion. On completion of that, the system displays the flower name that has been identified. The user is then invited to take another picture, if not the process can be completed. If no flow match occurs then the flow stops.



# **Experiments and Results**

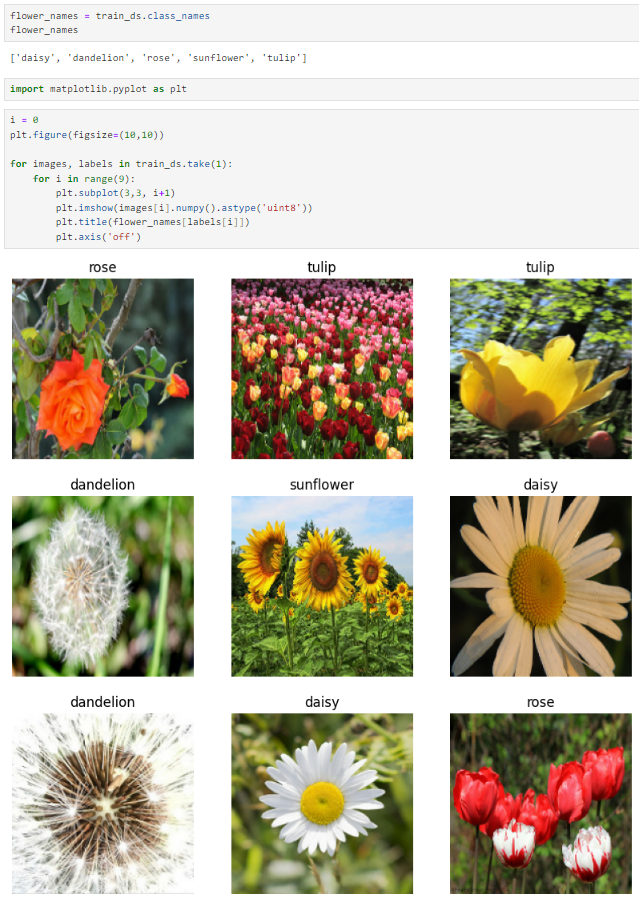
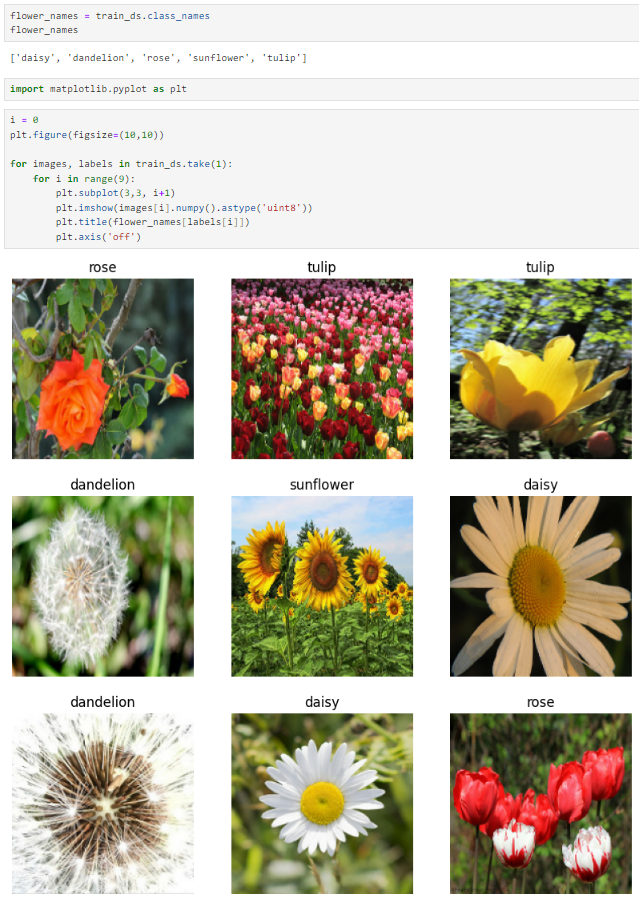
Using the capabilities of deep learning and attempting to differentiate between the different kinds of flowers using the data of 4317 images. The given dataset undergoes the image count and then split into training and validation sets, where there are 3454 training samples and 863 validation samples Data augmentation is used in this case where imagery is flipped, rotated, and zoomed in order to work for generalization. Thus, the visualization of the augmented dataset with the actual images contributes to the creation of a solid pipeline for model assessment on various variations of the data.



**Figure 1: Experiment 1** The code draws the list of subfolders in each subdirectory of the “Images” directory and prints the number of images in each subdirectory and the subdirectory’s name. It also informs the total images (4317 images) in the subfolder levels in order to present the overall structure of the dataset.



**Figure 2: Experiment 2** The code uses TensorFlow's image\_dataset\_from\_directory to split the images into training (3454 images) and validation (863 images) datasets, based on an 80-20 split. It sets the image size to 180x180 pixels and the batch size to 32, creating preprocessed datasets suitable for feeding into a machine learning model.



**Figure 3: Experiment 3** The train\_ds.class\_names returns a list of folder names in the dataset directory, corresponding to the categories: ['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']. The code then uses Matplotlib to draw 9 images in a 3x3 grid. The grid above represents the sample of 9 images selected from the dataset; each image is described with its class name (rose, tulip, dandelion and etc.). This visualization also validates and affirms that the dataset is correctly imported and labeled with five kinds of flowers.





**Figure 4: Experiment 4** A Sequential model is created to apply data augmentation on the images:

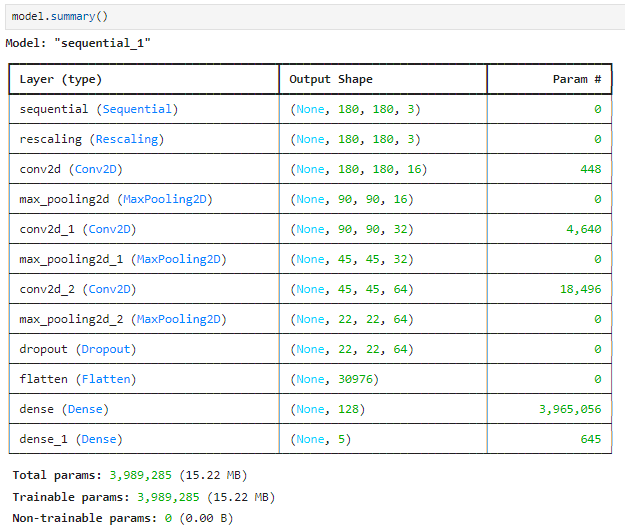
* **RandomFlip("horizontal")**: Randomly flips the images horizontally to create variability.
* **RandomRotation(0.1)**: Rotates the images by up to 10% of 360 degrees.
* **RandomZoom(0.1)**: Randomly zooms into the images by up to 10%.

The train\_ds element is then used to display a 3 by 3 grid of augmented image using ‘take(1)’ to select a single batch of images, the images are then passed through the data augmentation pipeline, and the results plotted using plt.subplot(3, 3, i + 1) and the axis labels turned off for clarity.

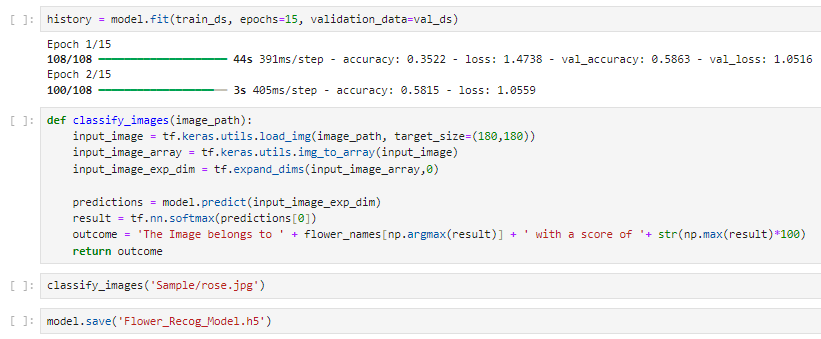
The images depicted above present the transformations resulting from data augmentation; for example, some of the dresses are inverted horizontally, rotated slightly to the right, or zoomed. The transformations produce variations in the training data set without modifying the image labels and, as such, enhances its capacity to generalize.



**Figure 5:** The image classification was accomplished using a convolutional neural network abbreviated CNN and built with TensorFlow’s Keras API. It includes three spatially convolutional layers each of which include a 2D convolution layer using ReLU actuation, and a pooling layer for dimensionality reduction. A dropout layer (rate = 0.2) was also included for regularization A flatten layer was included to transform feature maps to a one-dimensional vector. The network ends with two dense layers: one with 128 units (ReLU activation) and a final output layer with 5 units corresponding to the number of classes.



**Figure 6: Experiment 5** The output of this list shows the layer type, the output shape as well as the number of trainable parameters for each of the layers formed. Input images are of size 180×180×3 for all SCH networks and all SCH networks reduce spatial dimensions through max pooling and increase depth through filters over consecutive stages. The number of trainable parameters is 3,989,285. There is a dense layer with 128 neurons The output layer has 5 neurons for classification.



**Figure 7: Experiment 6**

**Model Training:** The model is further trained for 15 epochs by using fit function, where train\_ds is the training dataset and val\_ds is the validation dataset for the model. Epochs: on each of them accuracy and loss values for the training and validation sets are shown.

**Image Classification:** A function classify\_images is defined to predict the class of a single image.

* The image is loaded with a target size of 180×180.
* It is converted into a NumPy array and expanded to match the model's input shape.
* The model makes a prediction, and the output is processed using the softmax function to get class probabilities.
* The function returns the predicted class label and its confidence score.

**Testing and Saving:**

* The function classify\_images is called on a sample image (Sample/rose.jpg).
* The trained model is saved to a file named Flower\_Recog\_Model.h5 for later use.

# **Conclusion**

In this study, a convolutional neural network was developed to classify five flowers including daisy, dandelion, rose, sunflower and tulips, based on a dataset of 4317 images. The data set was divided into 80% training and 20% validation with 3454 training and 863 samples only. To further improve the capability of the model to generalize, some data augmentation techniques including flipping, rotation and zooming were performed.

Convolutional layers with ReLU activation and max pooling were used for feature extraction, with dropout for regularization; other layers included a flatten layer and two dense layers with five neurons in the final output layer. The network synthesized a substantial decrease in spatial dimensions with an increase in depth, and there were a total of 3,989,285 trainable parameters.

The model was trained for 15 steps and we can see that as it progresses the accuracy increases. A single image classifier function was created that gave the class label and confidence score for a single input image; the final model was also saved for use. In conclusion, the proposed deep learning pipeline provides a competitive solution for flower classification with the capability to learn new variation signatures thus generalizing the solution to unseen variation.

# **Recommendation**

For further enhancement of the NatureLens system the focus can be made on the improvement of the accuracy of detections as well as on the enlargement of the dataset, integration with IoT devices for real-time monitoring of the situation, on the further development of the mobile application with such functions as augmented reality and multilingual support. Also, it is planned to expand using other types of machine learning algorithms and consider the opinions of users to improve the performance and convenient application.

##### VII. **Acknowledgment**

First and foremost, we express our profound thankfulness to Almighty God for His constant direction, wisdom, and strength over the course of this research. His blessings and provision have served as both our basis and source of inspiration.

We also extend our heartfelt gratitude to our instructor, Engr. Bernard Yasay, constructive feedback, and encouragement, which have significantly improved our research. His dedication and expertise have had an impact on our knowledge and approach to this project.

Finally, we extend sincere appreciation to our families for their constant support, patience, and love. The support they provide has been our driving force, inspiring us to strive and succeed.

##### VIII. **References**

1. S. Chavez, “The Future of Gardening: Exploring the Potential of Smart Plant Sensors,” The Connected Shop, Apr. 16, 2024. https://theconnectedshop.com/blogs/tech-talk/the-future-of-gardening-exploring-the-potential-of-smart-plant-sensors?[srsltid=AfmBOoqJ4y7A9CDa\_ELuXd26mcfQHLY3bYVo4S2X9wsHCk1lUiTQGTka](https://theconnectedshop.com/blogs/tech-talk/the-future-of-gardening-exploring-the-potential-of-smart-plant-sensors?srsltid=AfmBOoqJ4y7A9CDa_ELuXd26mcfQHLY3bYVo4S2X9wsHCk1lUiTQGTka)
2. Shivrai Technologies Pvt. Ltd. “How Agriculture Software Boosts Sustainable Farming Practices,” FarmERP, Nov. 05, 2023. https://www.farmerp.com/how-a-plant-nursery-management-system-can-streamline-inventory-control
3. J. Wäldchen and P. Mäder, “Plant Species Identification Using Computer Vision Techniques: A Systematic Literature Review,” Archives of Computational Methods in Engineering, vol. 25, no. 2, pp. 507–543, Jan. 2017, doi: https://doi.org/10.1007/s11831-016-9206-z.
4. TensorFlow. (n.d.). TensorFlow. https://www.tensorflow.org/about/bib
5. Streamlit: A faster way to build and share data apps. (n.d.). https://streamlit.io/