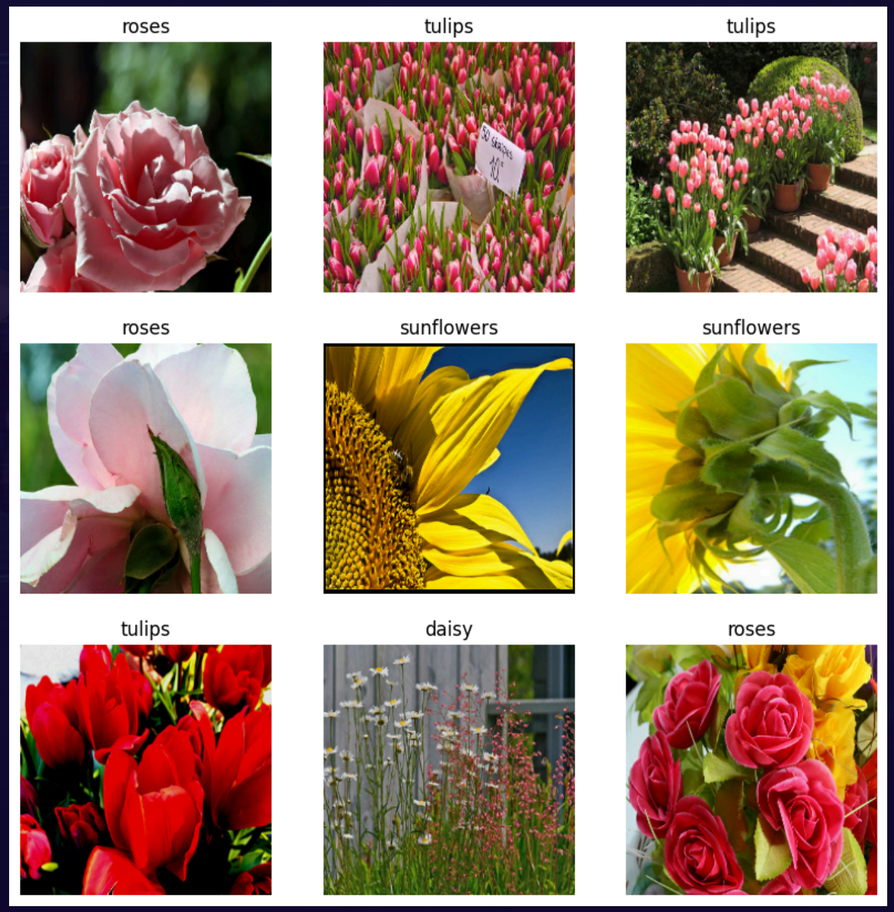
Flowers Classification

Project created by
866110 - Simone Adobati
866453 - Alisia Sallemi
869008 - Andrea D'Amicis

Project Overview

The goal of the project is to create a Deep Learning Neural Network able to classify 5 different classes of flowers: Daisies, Dandelions, Roses, Sunflowers and Tulips.



Chosen Dataset

The dataset we chose contains a total of 3670 images:

- 633 daisies
- 898 dandelions
- 641 roses
- 699 sunflowers
- 799 tulips

All the instances have different sizes, brightness, saturation and angles.

Data Acquisition

All the images are collected and divided in three different sets, training, validation and test. They are resized to 224x244 in order to have the same amount of pixels in each image.
The proportions are 70% training, 15% validation and 15% test.
The same process is done another time applying a grayscale process to the images.

Architectures

Max accuracy
71.64%

Architecture 1

- Input layer - 224, 224, 3
- Conv2D - Filter 16, Kernel 3
- Conv2D - Filter 32, Kernel 3
- Dense - 16 neurons
- Output layer - 5 neurons

Optimizer: Adam(LR: 0.001)

Max accuracy
71.45%

Architecture 2

- Input layer - 224, 224, 3
- Conv2D - Filter 16, Kernel 5
- Conv2D - Filter 32, Kernel 5
- Dense - 16 neurons
- Output layer - 5 neurons

Optimizer: Adam(LR: 0.001)

Max accuracy
72.36%

Architecture 3

- Input layer - 224, 224, 3
- Conv2D - Filter 16, Kernel 3
- Conv2D - Filter 16, Kernel 3
- Conv2D - Filter 32, Kernel 3
- Dense - 32 neurons
- Dense - 16 neurons
- Dense - 8 neurons
- Output layer - 5 neurons

Optimizer: Adam(LR: 0.001)

WITHOUT: RandomFlip, RandomRotation, RandomContrast, RandomBrightness

WITH: RandomFlip, RandomRotation, RandomContrast, RandomBrightness

Keras Hyperband

In order to find the optimal model, based on the third basic architecture, we used a tool from the keras_tuner library called Hyperband.
This object implement an hyperparameter optimization algorithm. The idea behind it, is to iteratively train models with different configurations for a few epochs and only continue training the best-performing models for more epochs.

Keras Hyperband

We fixed the number of layers to 3 Convolutional layers with kernel size 3 and 3 Dense layers (in addition to an Input layer 224, 224, 3 and an Output layer with 5 neurons).
We let the Hyperband find the optimal number of filters and neurons in each layer, choosing between these numbers:

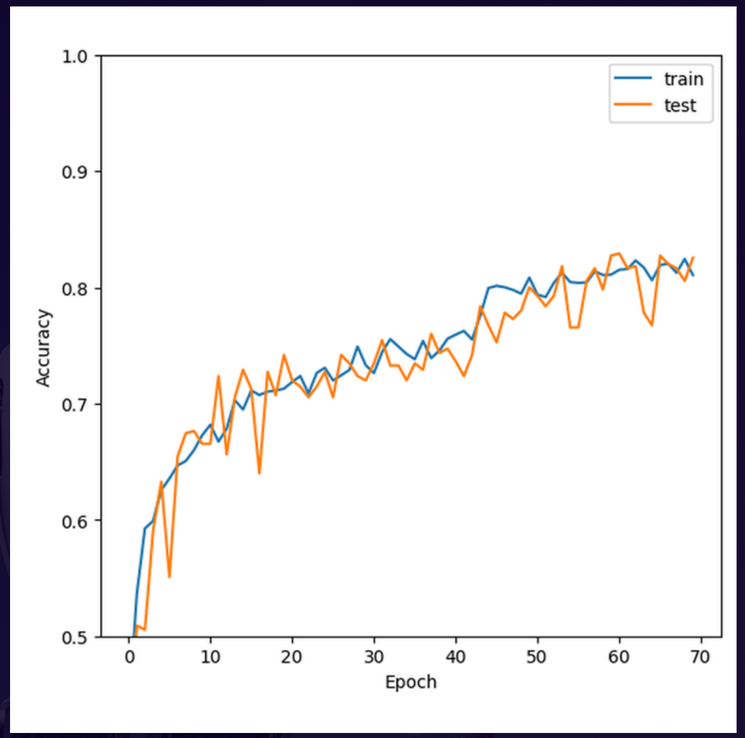
1. **Conv2D - 32 or 64**
2. **Conv2D - 64 or 128**
3. **Conv2D - 64 or 128**
4. **Dense - 64 or 128**
5. **Dense - 32 or 64**
6. **Dense - 16 or 32**

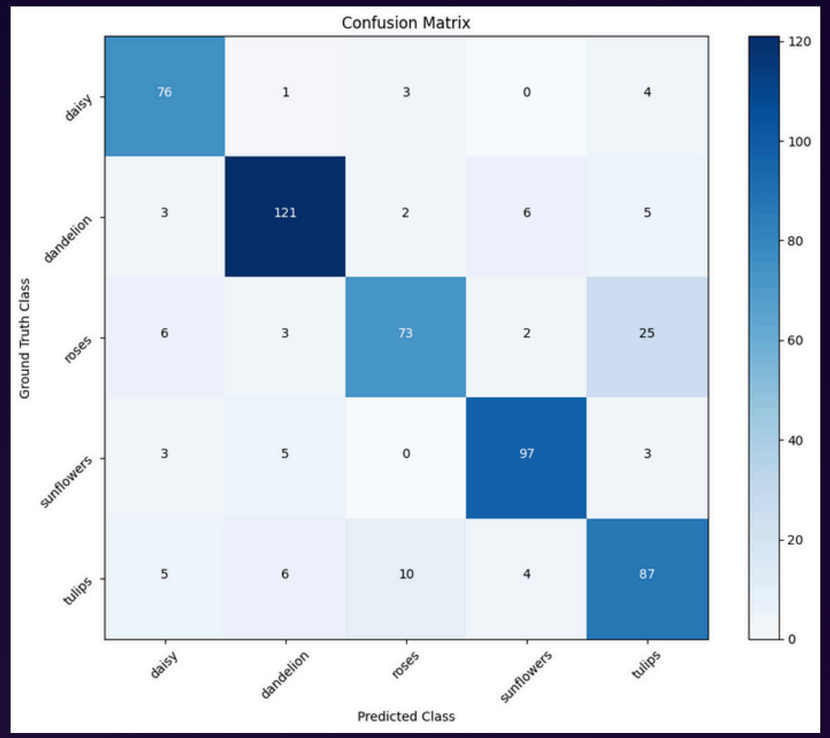
Best val_accuracy: 0.76
Total elapsed time: 23h 39m 55s

units_conv_1: 64
units_conv_2: 64
units_conv_3: 128
units_dense_1: 64
units_dense_2: 64
units_dense_3: 16
tuner/epochs: 60
tuner/initial_epoch: 20
tuner/bracket: 2
tuner/round: 2
tuner/trial_id: 0068

Final Model

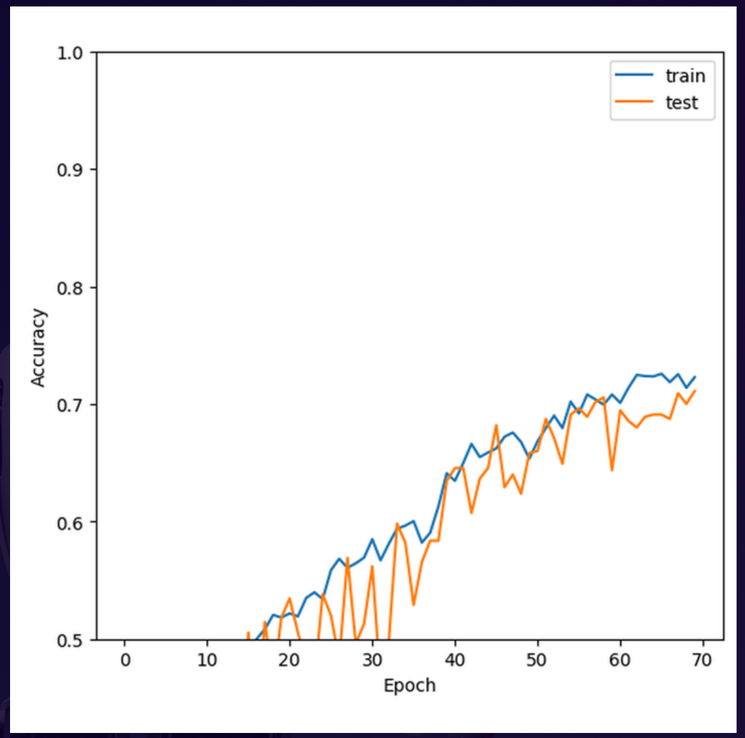
After finding out the best model provided by the Hyperband, we trained the network and we fitted it over 70 epochs, reaching a final accuracy of 82.55%, with a max of 82.91%. We used the "ReduceLROnPlateau" callback from the keras library in order to update the learning rate.

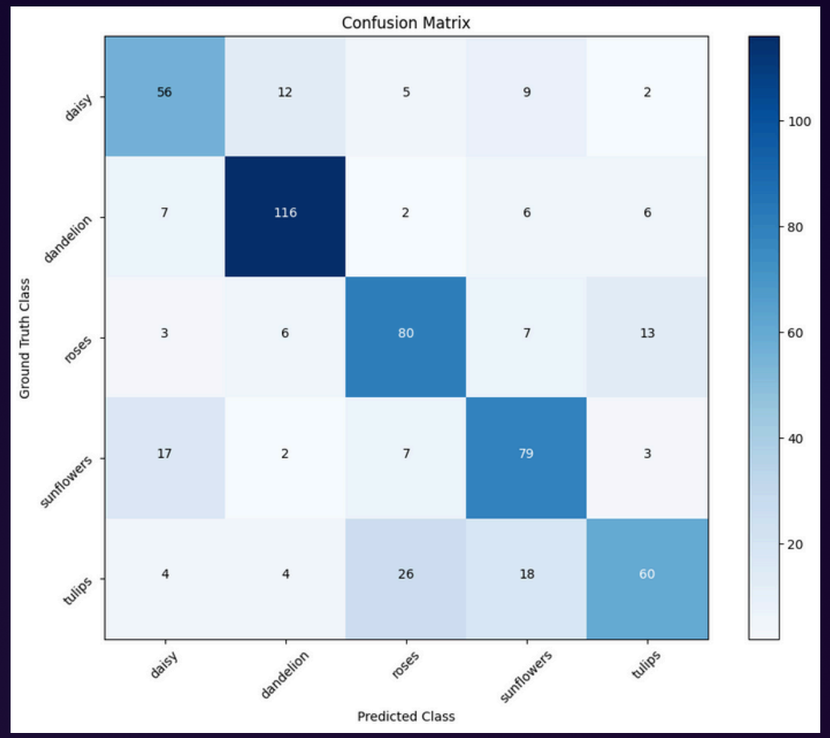




Final Model Grayscale

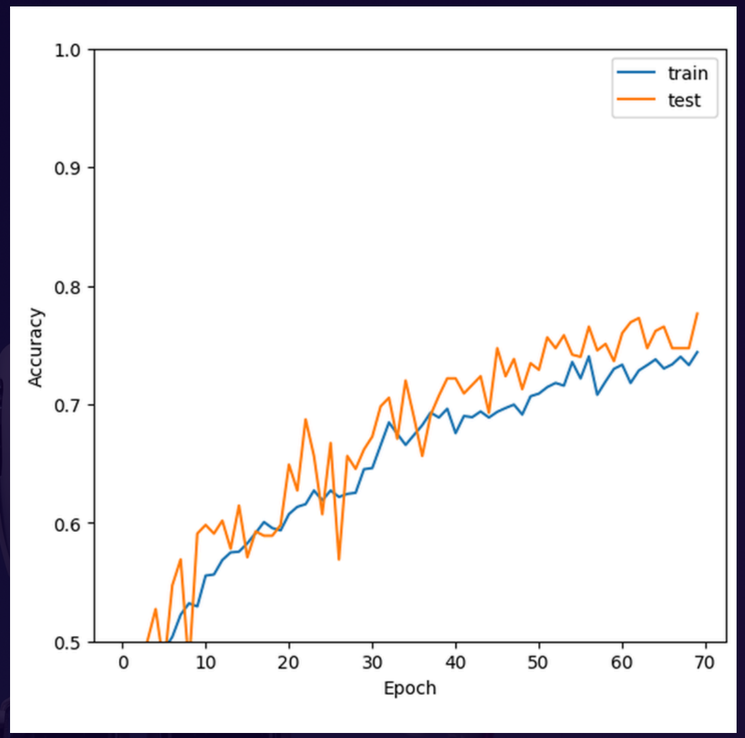
We also tried the same architecture but with grayscale images, the only change we made was setting the Input layer to (224, 224, 1), because the gray scale only has 1 channel for the color. The model was fitted over 70 epochs, reaching a final accuracy of 71.09%, that is also the max.

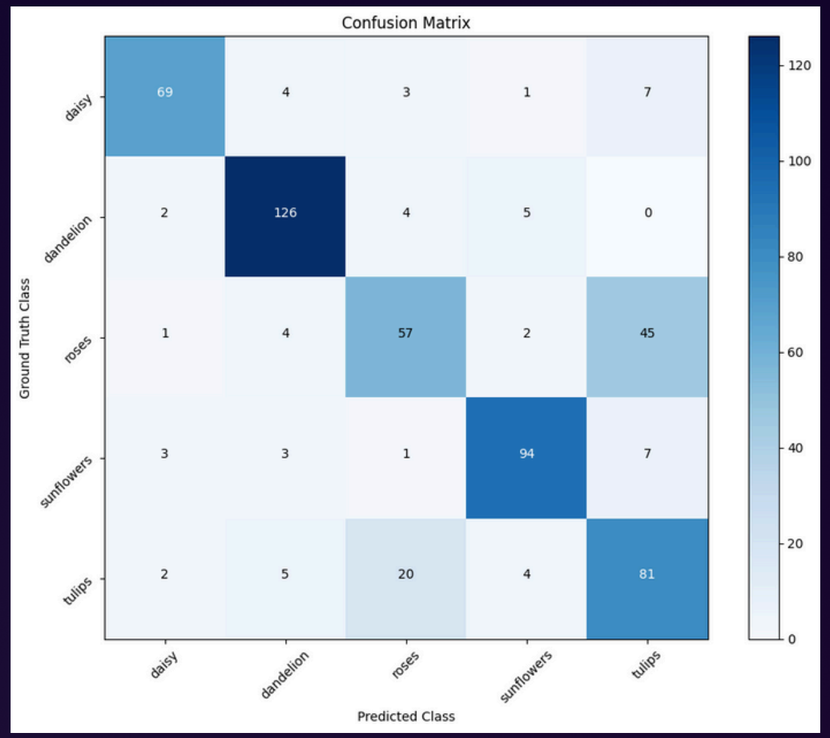




Final Model with Dropout

We added a model with dropout layers in the Dense layers section and the result, after fitting over 70 epochs, is a final accuracy of 77.64%, that is also the max.





Pre-trained models

Max accuracy
62.55%

ResNet50V2

This architecture present a lot of overfitting, it has a difference between training and test accuracy of 17.07%.

Max accuracy
92.73%

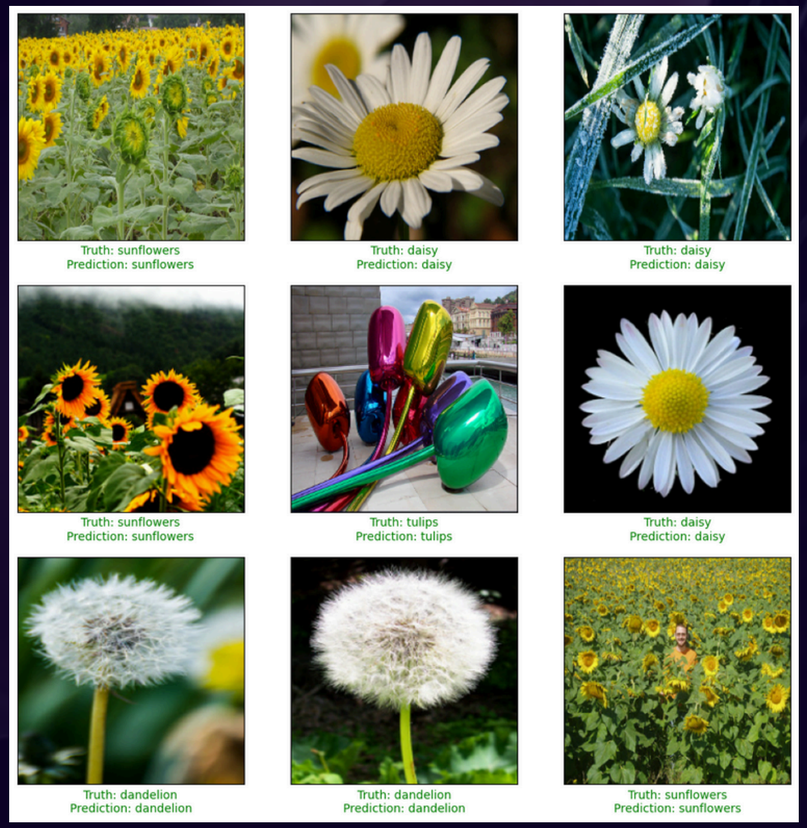
EfficientNetB3

This architecture reaches 100% accuracy on the training in 13 epochs, the test set reaches an high value of accuracy but still remain a bit far from the training set.

- Worst than our best model
- Better than our best model

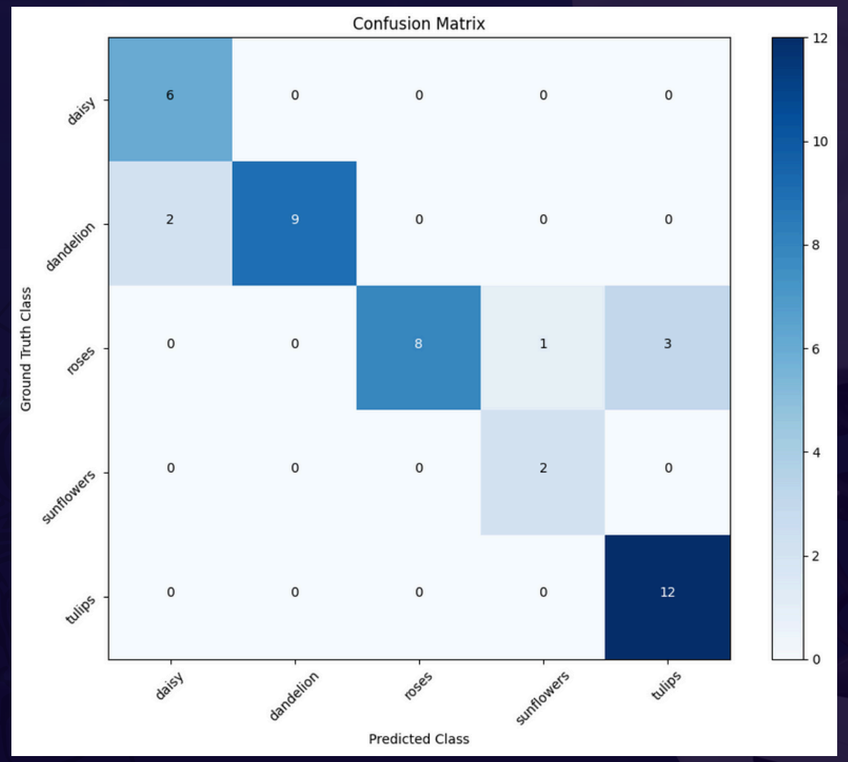
Testing the model

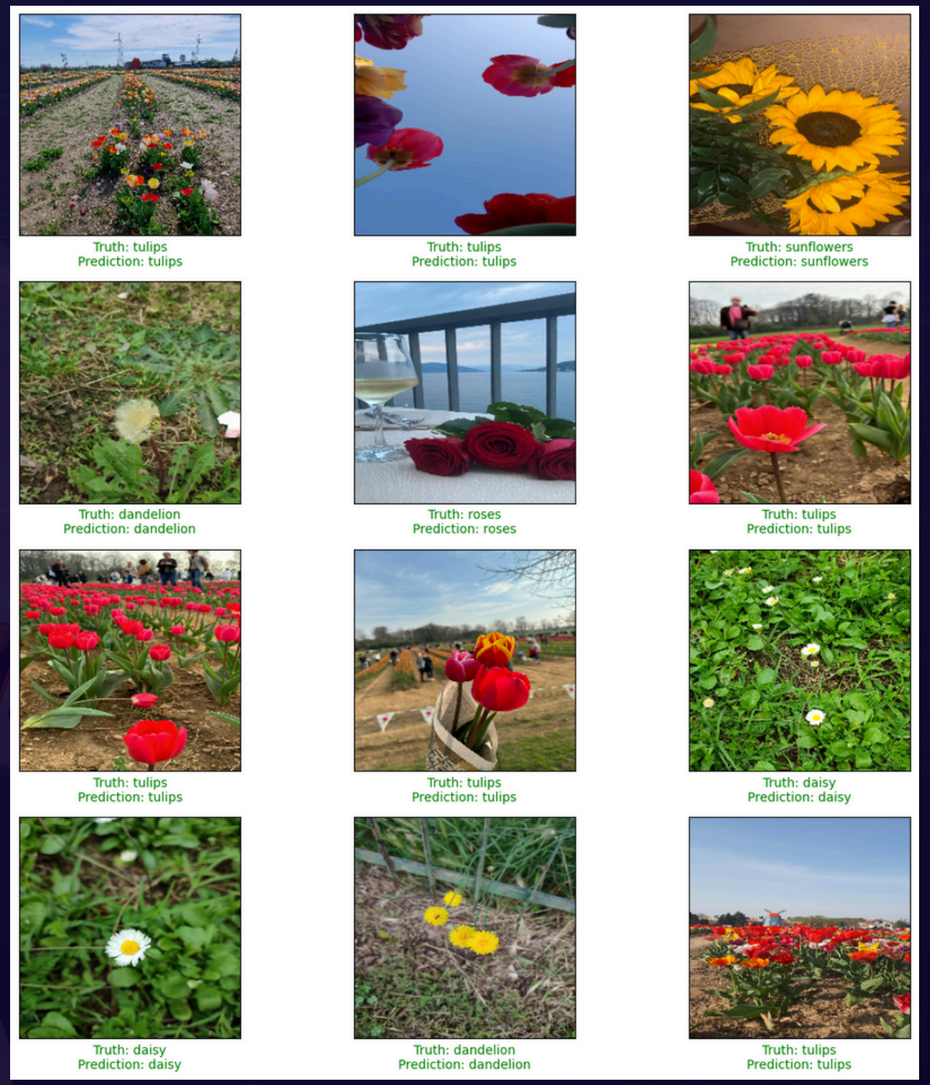
Using our best model, we wanted to see which were the correctly and incorrectly classified images, in order to find out if the errors were made over difficult images or not.



Testing our images

We also tried the model over additional images of flowers we found around the city. We used 43 images and the architecture correctly classified 37 of them, reaching an accuracy of 86%.



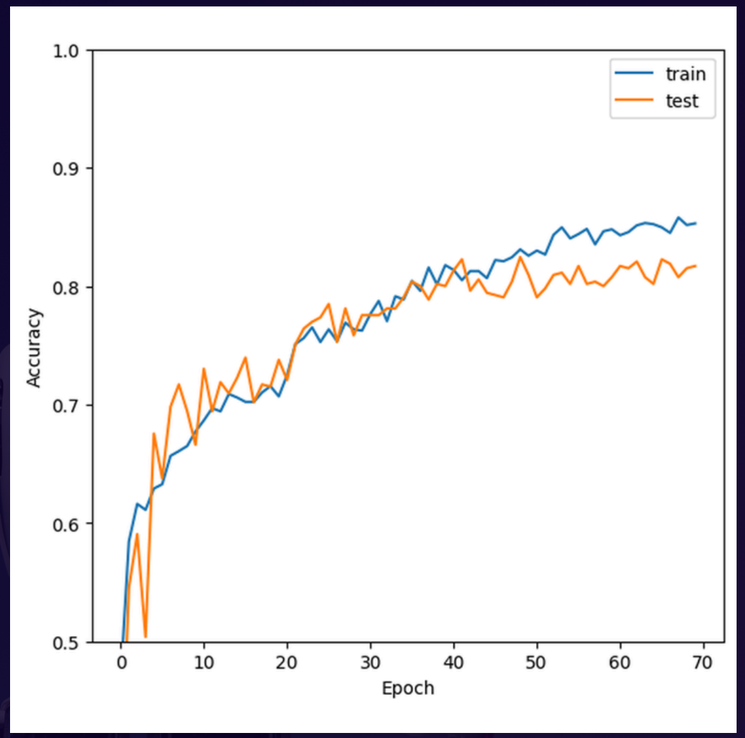


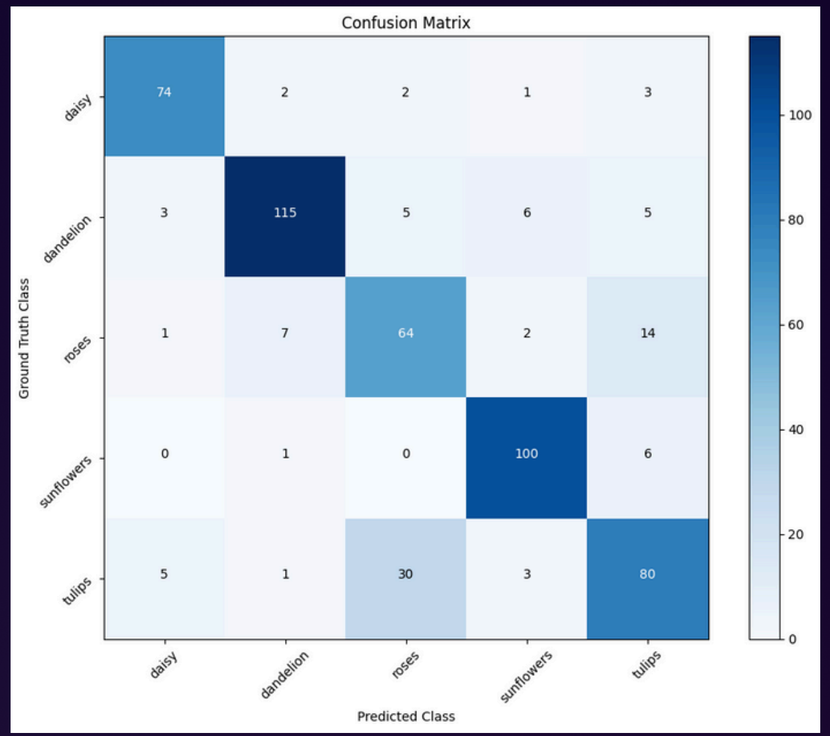
Data Cleaning

Maybe one way to try to increase the accuracy of the model is to improve the quality of the data. We trained a model using a cleaned version of the dataset to see if our hypothesis was true.

Model with Cleaned Data

The results we obtained are not the one expected. The final accuracy is 81.70%, having a max accuracy of 82.26%. This can mean that the quantity of misleading images in the dataset is minimal compared to the total amount of instances.






Thank You!