Flowers recognition: CNN InceptionV3 model

SUBHAJIT CHATTERJEE

AD20216803

Table of Contents

Introduction

- Concept of image classification :
  - Image classification, core task in computer vision.
  - In order to classify a set of data into different classes or categories, the relationship between the data and the classes into which they are classified must be well understood.
  - To achieve this by computer, the computer must be **trained.**
  - Training Is key to the success of classification.
  - Classification techniques were originally developed out of research in **Pattern Recognition** field.
  - Classification of remotely sensed images involves the process of the computer program learning the relationship between the **data** and the information **classes**.
  - Important aspects of accurate classification
    - **Learning techniques**
    - **Feature sets**

Related Work

Ref	Description	Data set used	Finding	Algorithms	Outcomes
1 .	Robust Sports Image Classification Using InceptionV3 and Neural Networks	The dataset is made by taking videos of each sports category from YouTube and then frames are extracted from each video.	A robust framework for classifying the sport images based on the environment and related surroundings.	Random forest, KNN, and SVM	Achieved an average accuracy of 96.64%.
2 .	Inception v3 based cervical cell classification combined with artificially extracted features	The Herlev dataset, this dataset only includes 917 images.	The cell features can be extracted automatically, the cervical cell domain knowledge will be lost, and the corresponding features of different cell types will be missing. hence, the classification effect is not sufficiently accurate.	Combines Inception v3 and artificial features.	Accuracy of more than 98% is achieved.
3 .	Towards more efficient CNN-based surgical tools classification using transfer learning	Cholec80 laparoscopy video dataset.	We implement a fine-tuned CNN to tackle the automatic tool detection during a surgery, with prospective use in the teaching field, evaluating surgeons, and surgical quality assessment (SQA).	InceptionResnet-v2	A mean average precision of 93.75%.

Dataset

Images are sorted into folders according to classes : Images in this dataset are divided into five categories. ['dandelion', 'daisy', 'sunflower', 'tulip', 'rose']

This Data contains around 4317 images of size 150 x 150 x 3 distributed under 5 categories.

‘dandelion' -> 0,

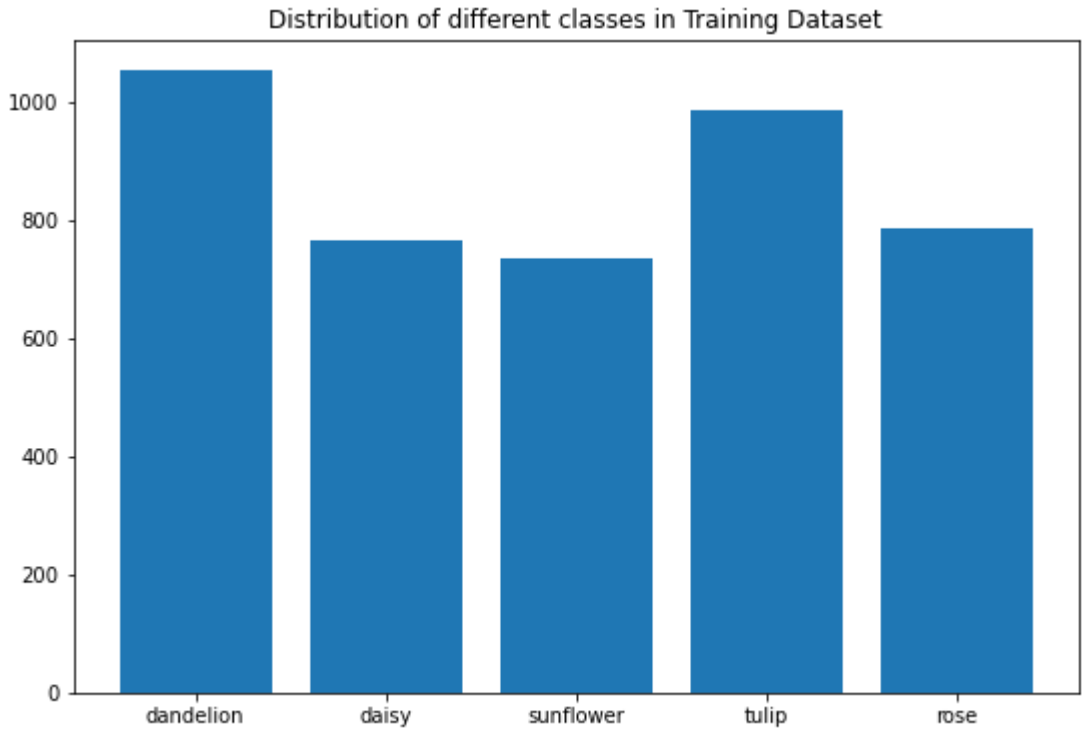
‘daisy' -> 1,

‘sunflower' -> 2,

‘tulip' -> 3,

‘rose' -> 4

Distribution of different classes in Training Dataset



Randomly sample one image per class

Class: dandelion

Class: daisy

Class: sunflower

Class: tulip

Class: rose

Label encoding

- We have 5 classes of species and 5 labels for each of them (0,1,2,3,4).

```
# Convert classes in digit form
num_classes = len(np.unique(y))
print(f'Classes: {num_classes} and corresponding labels: {labels}')
```

```
Classes: 5 and corresponding labels: Int64Index([0, 1, 2, 3, 4], dtype='int64')
```

- In order to pass them on network inputs we should make some preparation known as One-Hot Encoding, which takes a single integer and produces a vector where a single element is 1 and all other elements are 0, like [0, 1, 0, 0, 0].
- There are several ways in Python to do it, we will choose Keras's to_categorical() in our implementation.

Data Summary Table

Sl no.	Class	Number of images	No. of images for Training (60%)	No. of images for Validation (10%)	No. of images for Testing (30%)
0	dandelion	1052	631	105	316
1	daisy	764	458	76	229
2	sunflower	733	440	73	220
3	tulip	984	591	98	295
4	rose	784	470	78	236
	Total	4317	2590	431	1296

Model

a) **InceptionV3** model which is one of the ImageNet winners with very high accuracy and low computer resources.

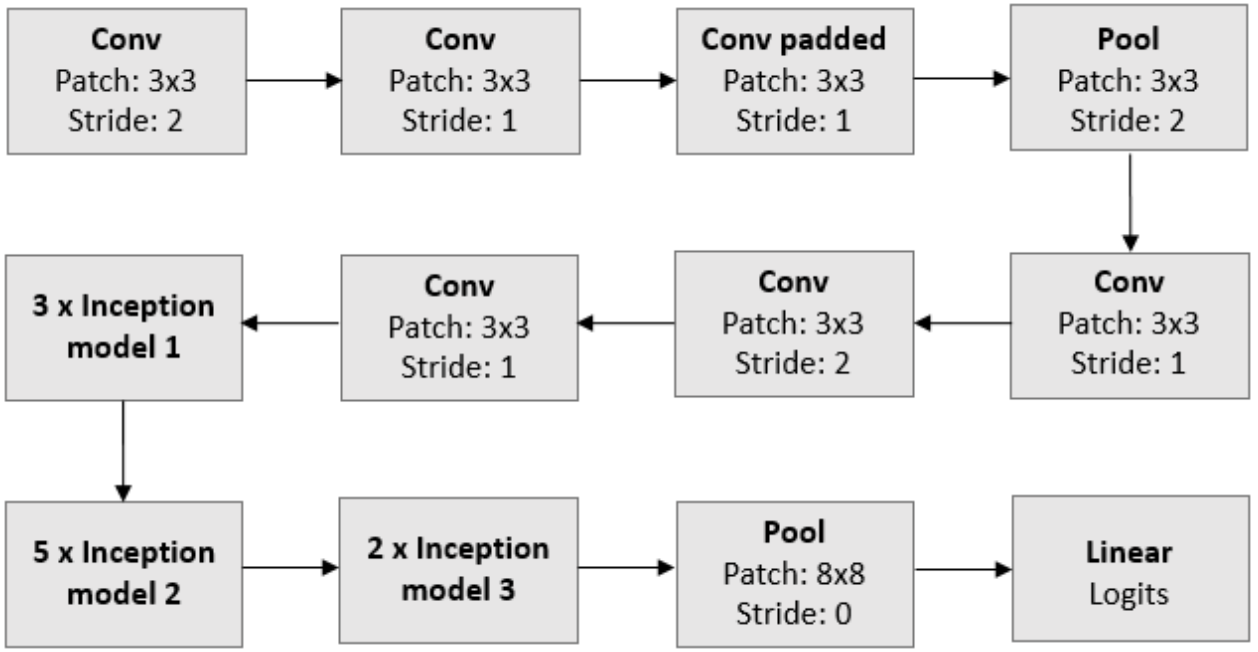
b) Use a Transfer Learning paradigm.

c) **InceptionV3** mainly focuses on burning less computational power by modifying the previous Inception architectures.

- Transfer learning refers to a technique for predictive modeling on a different but somehow similar problem that can then be reused to accelerate the training and improve the performance of a new model.
- It becomes clear that on the first layers network is trying to grasp the most basic laws (edges), on the middle layers - the most important (shapes), and on the last layer - **specific details** (high level features).
- In deep learning, this means reusing the weights in one or more layers from a pre-trained network model in a new model and either keeping the weights fixed, fine tuning them, or adapting the weights entirely when training the model
- We can use pre-trained model and train it with a **small set of our data**. The trick is here to freeze or lock the early layers and let the final layers to be trained. In this way, our new model can know the facial patterns in middle and high level features as well.

InceptionV3 layer architecture

type	patch size/stride or remarks	input size
conv	3×3/2	299×299×3
conv	3×3/1	149×149×32
conv padded	3×3/1	147×147×32
pool	3×3/2	147×147×64
conv	3×3/1	73×73×64
conv	3×3/2	71×71×80
conv	3×3/1	35×35×192
3×Inception	As in figure 5	35×35×288
5×Inception	As in figure 6	17×17×768
2×Inception	As in figure 7	8×8×1280
pool	8 × 8	8 × 8 × 2048
linear	logits	1 × 1 × 2048
softmax	classifier	1 × 1 × 1000

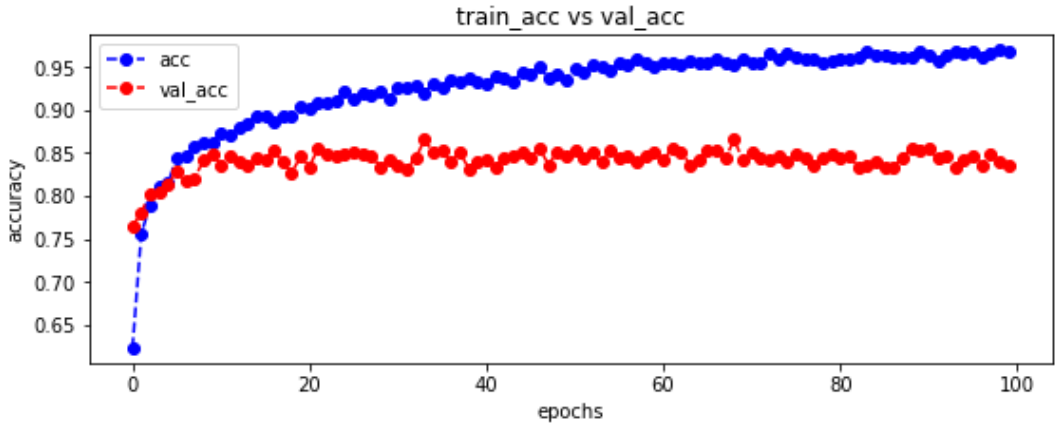


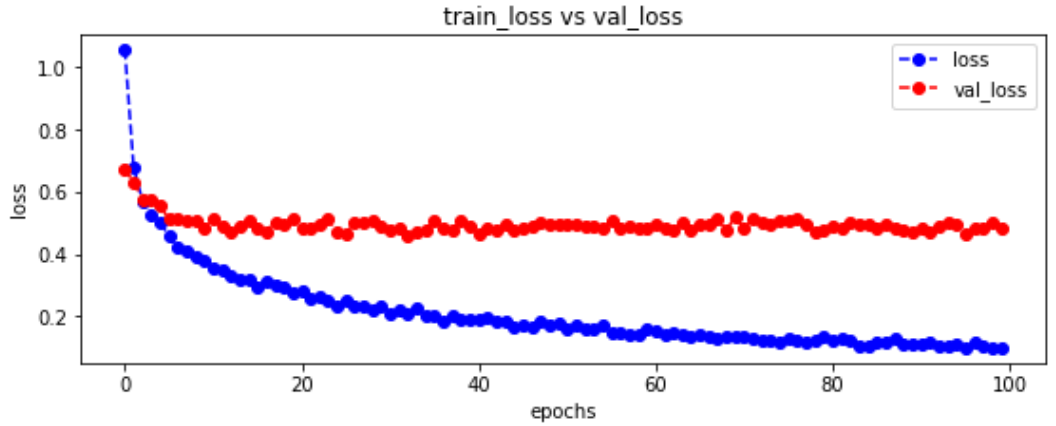
Result

```
161/161 [==============================] - 14s 89ms/step - loss: 0.1025 - accuracy: 0.9697 - val_loss: 0.4692 - val_accuracy: 0.8534
Epoch 95/100
161/161 [==============================] - 15s 94ms/step - loss: 0.1032 - accuracy: 0.9662 - val_loss: 0.4613 - val_accuracy: 0.8630
Epoch 96/100
161/161 [==============================] - 15s 90ms/step - loss: 0.1042 - accuracy: 0.9685 - val_loss: 0.4698 - val_accuracy: 0.8582
Epoch 97/100
161/161 [==============================] - 15s 94ms/step - loss: 0.0967 - accuracy: 0.9685 - val_loss: 0.5031 - val_accuracy: 0.8462
Epoch 98/100
161/161 [==============================] - 14s 89ms/step - loss: 0.1033 - accuracy: 0.9697 - val_loss: 0.4892 - val_accuracy: 0.8462
Epoch 99/100
161/161 [==============================] - 15s 91ms/step - loss: 0.0993 - accuracy: 0.9670 - val_loss: 0.4674 - val_accuracy: 0.8582
Epoch 100/100
161/161 [==============================] - 15s 92ms/step - loss: 0.1031 - accuracy: 0.9678 - val_loss: 0.4915 - val_accuracy: 0.8558
Time : 0:24:44.238866
```

Time taken for training with 100 epochs

We Trained our model with 100 epochs





Training and Validation (accuracy and loss) graph

Recognition rate

```
# Evaluate on test dataset
loss, acc = inception_model.evaluate_generator(test_generator, verbose=0)
print(f'Test loss: {loss:.2f}%')
print(f'Recognition rate : {acc*100:.2f}%')
```

```
Test loss: 0.46%
Recognition rate : 84.95%
```

Confusion matrix

Support set

245	[[198 16 6 21 4]
311	[7 291 3 6 4]
215	[4 4 174 7 26]
213	[2 12 6 180 13]
312	[6 10 29 9 258]]

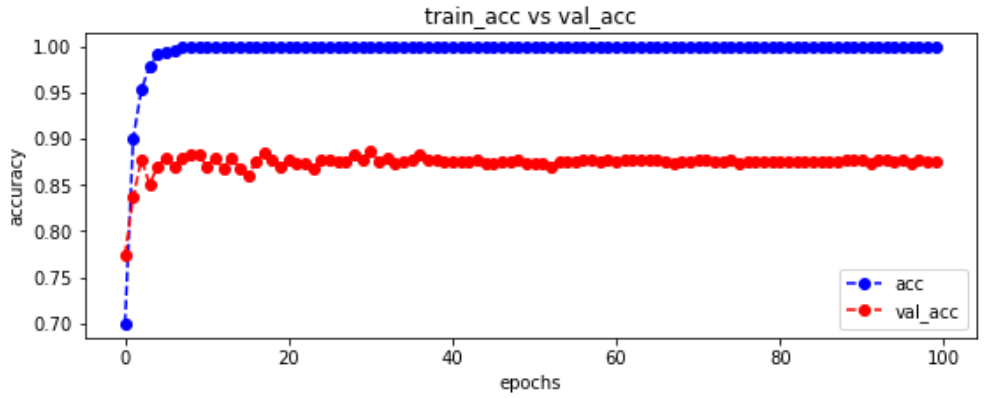
Total 195 images misclassified as another class.

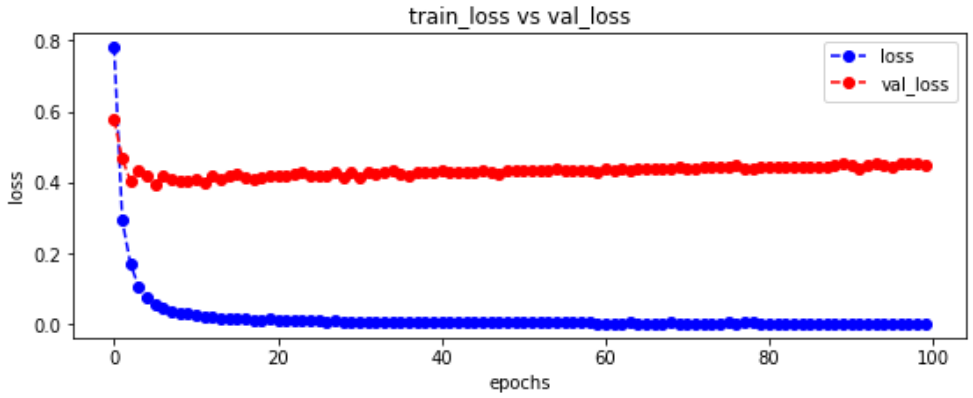
Changed parameter and image size (224 x 224 x 3)

```
Epoch 95/100
161/161 [==============================] - 7s 46ms/step - loss: 0.0030 - accuracy: 0.9996 - val_loss: 0.4478 - val_accuracy: 0.8750
Epoch 96/100
161/161 [==============================] - 7s 46ms/step - loss: 0.0034 - accuracy: 0.9992 - val_loss: 0.4449 - val_accuracy: 0.8774
Epoch 97/100
161/161 [==============================] - 7s 46ms/step - loss: 0.0036 - accuracy: 0.9996 - val_loss: 0.4563 - val_accuracy: 0.8726
Epoch 98/100
161/161 [==============================] - 7s 45ms/step - loss: 0.0025 - accuracy: 0.9996 - val_loss: 0.4527 - val_accuracy: 0.8774
Epoch 99/100
161/161 [==============================] - 7s 46ms/step - loss: 0.0046 - accuracy: 0.9992 - val_loss: 0.4537 - val_accuracy: 0.8750
Epoch 100/100
161/161 [==============================] - 7s 46ms/step - loss: 0.0028 - accuracy: 0.9996 - val_loss: 0.4508 - val_accuracy: 0.8750
Time : 0:12:33.602899
```

Time taken for training with 100 epochs

We Trained our model with 100 epochs





Training and Validation (accuracy and loss) graph

Recognition rate

```
# Evaluate on test dataset
loss, acc = inception_model.evaluate_generator(test_generator, verbose=0)
print(f'Test loss: {loss:.2f}%')
print(f'Recognition rate : {acc*100:.2f}%')
```

```
Test loss: 0.52%
Recognition rate : 85.57%
```

Confusion matrix

Total 187 images misclassified as another class.

Support set

245	[[203 12 4 18 8]
311	[4 283 2 16 6]
215	[2 4 179 4 26]
213	[4 11 7 179 12]
312	[4 13 22 8 265]]

Conclusion

- We solved a flower classification problem using Machine Learning method - a CNN (Inception v3 model) using Transfer Learning approach with 85.57% accuracy.
- During Machine Learning process we divided flowers dataset into three parts: train (60%), validation (10%) and test(30%).
- First shown model validation data and then made final prediction on test. This is a classical approach. Howewer, accuracy can be increased using more epochs or tuning model's architecture or/and its hyper parameters.

References

- https://www.sciencedirect.com/science/article/pii/S1568494620302519
- https://www.sciencedirect.com/science/article/pii/S1877050920307560
- https://link.springer.com/article/10.1186/s40537-021-00509-8
- https://arxiv.org/abs/1512.00567
- https://machinelearningmastery.com/how-to-improve-performance-with-transfer-learning-for-deep-learning-neural-networks/
- https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/

THANK YOU