Image Classification by Convolutional Neural Networks

Aung Hein

18 Jun 2024

Outline

- Introduction
- Data preprocessing
- VGG16
- InceptionV3
- ResNet152V2
- Conclusion
- Appendix

Introduction

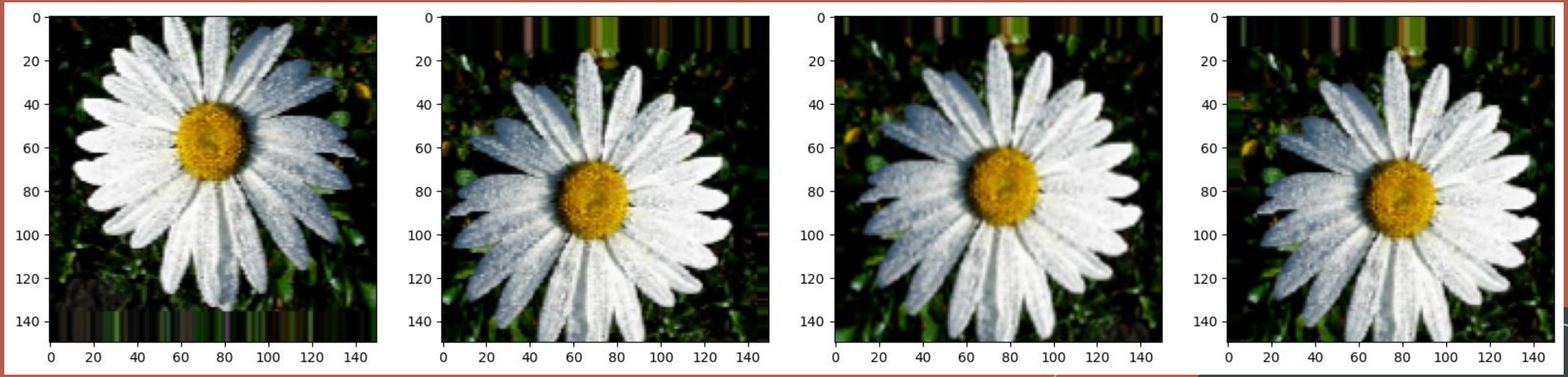
- In this project, I do experiment about few pretrained Convolutional Neural Network for image classification
- I been tested on VGG16, Inception V3 and ResNet152V2
- I used flower images dataset from TensorFlow

Dataset link:

- https://www.tensorflow.org/tutorials/load_data/images

Data Preprocessing

- There are 5 classes on image dataset : ['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']
- Image data integer was rescaled by dividing with 255
- 3457 train images and 860 validation images was split, generate variations and prepared by ImageDataGenerator class



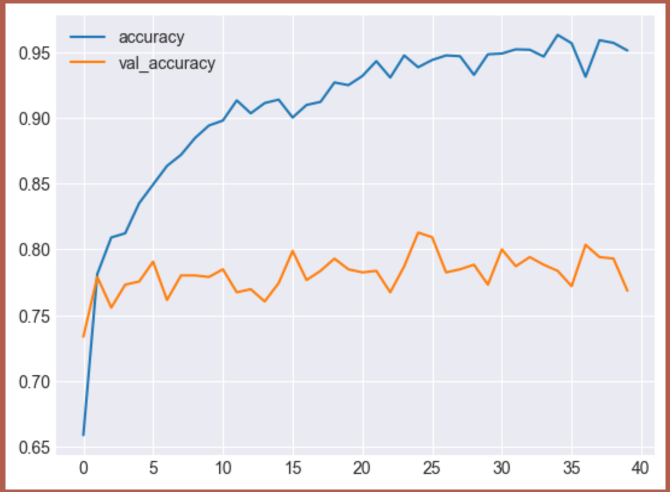
Visual Geometry Group (VGG16)

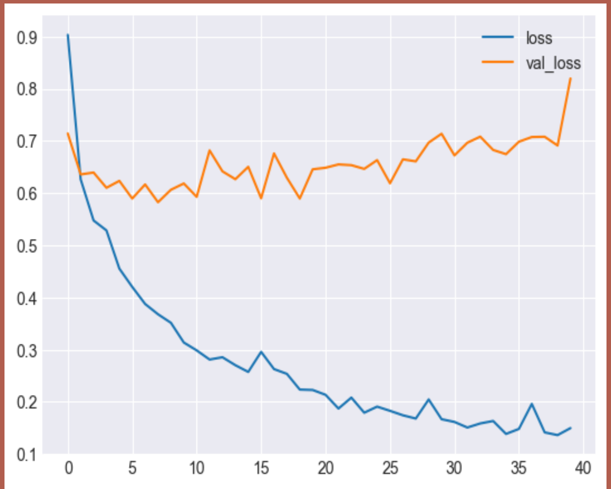
VGG16

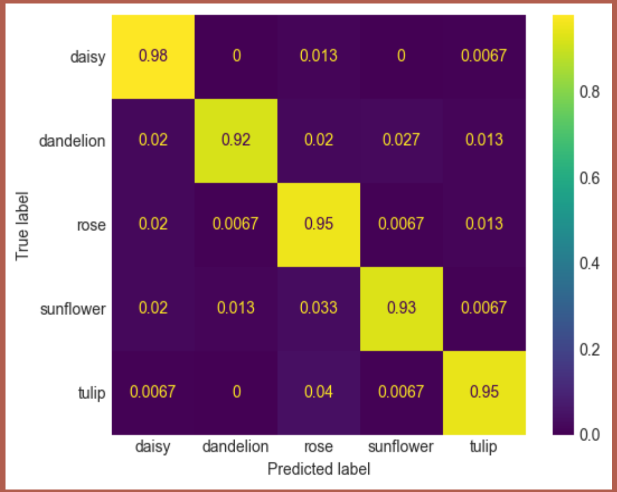
Optimizer :

'adam'

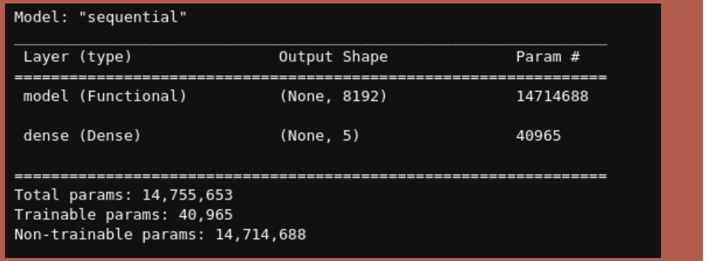
val_accuracy: 0.7686







True label \ Predicted label	daisy	dandelion	rose	sunflower	tulip
daisy	0.98	0	0.013	0	0.0067
dandelion	0.02	0.92	0.02	0.027	0.013
rose	0.02	0.0067	0.95	0.0067	0.013
sunflower	0.02	0.013	0.033	0.93	0.0067
tulip	0.0067	0	0.04	0.0067	0.95



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 model (Functional)          (None, 8192)              14714688

 dense (Dense)               (None, 5)                 40965

=================================================================
Total params: 14,755,653
Trainable params: 40,965
Non-trainable params: 14,714,688
```

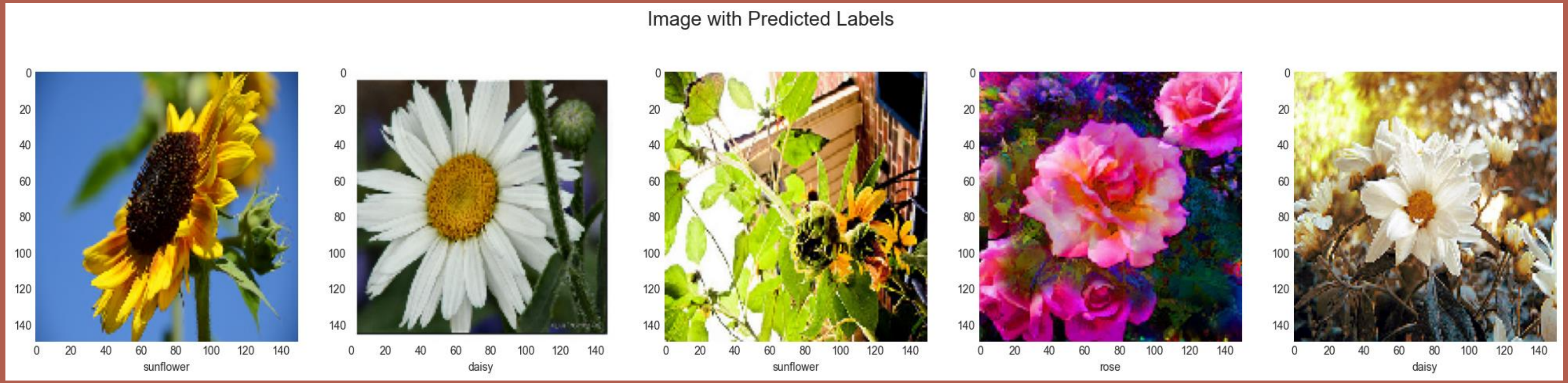


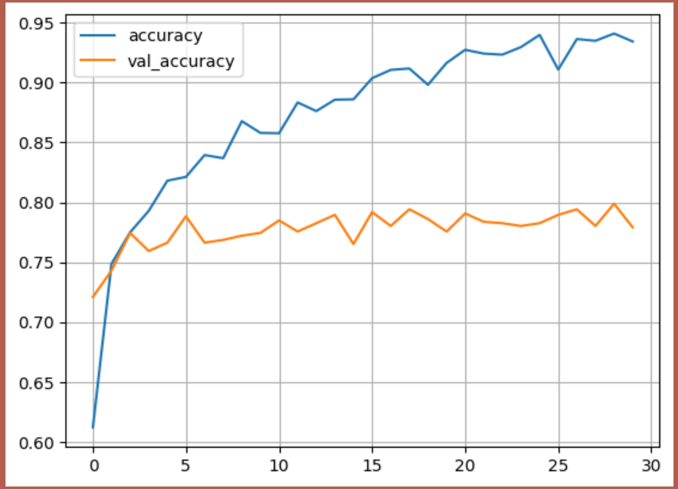
Image with Predicted Labels

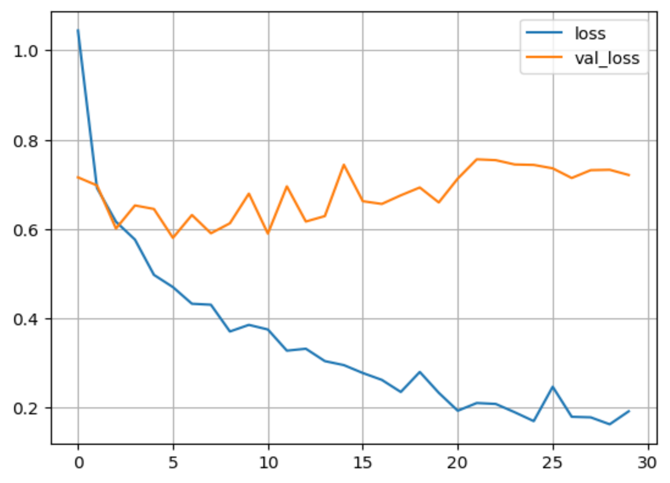
VGG16 with additional dense layer

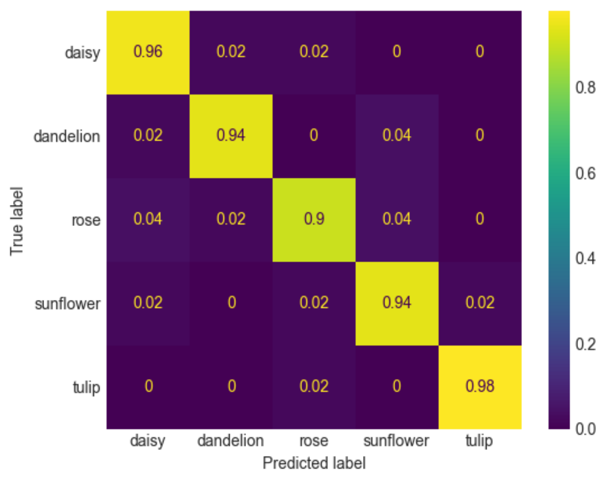
Optimizer :

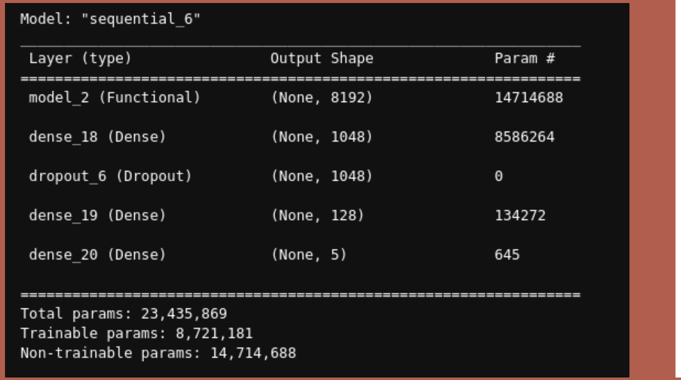
'adam'

val_accuracy: 0.7791

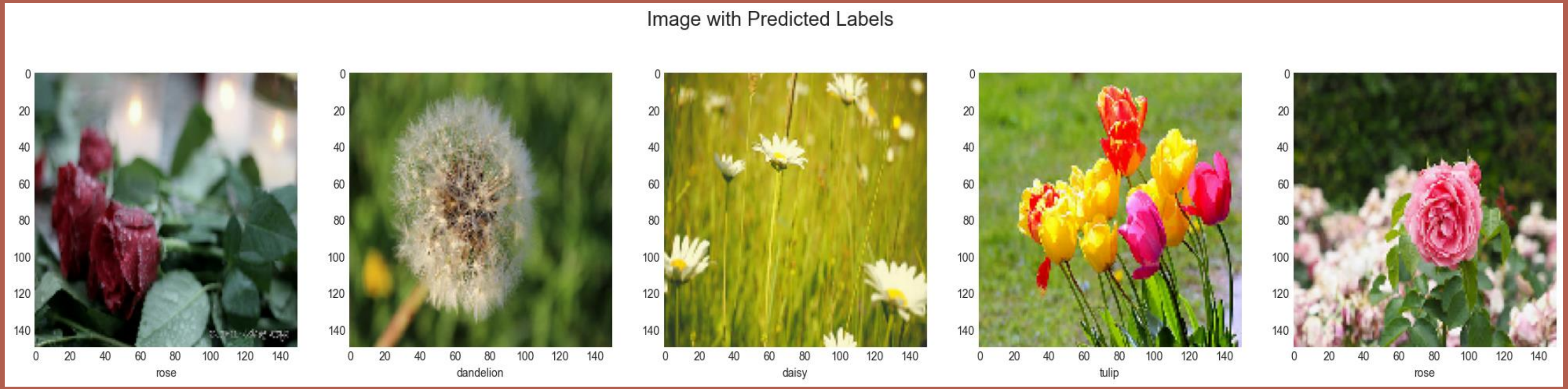








```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 model_2 (Functional)        (None, 8192)              14714688

 dense_18 (Dense)            (None, 1048)              8586264

 dropout_6 (Dropout)         (None, 1048)              0

 dense_19 (Dense)            (None, 128)               134272

 dense_20 (Dense)            (None, 5)                 645

=================================================================
Total params: 23,435,869
Trainable params: 8,721,181
Non-trainable params: 14,714,688
```



VGG16 with Tuning

Optimizer :

SGD (lr = 0.001)

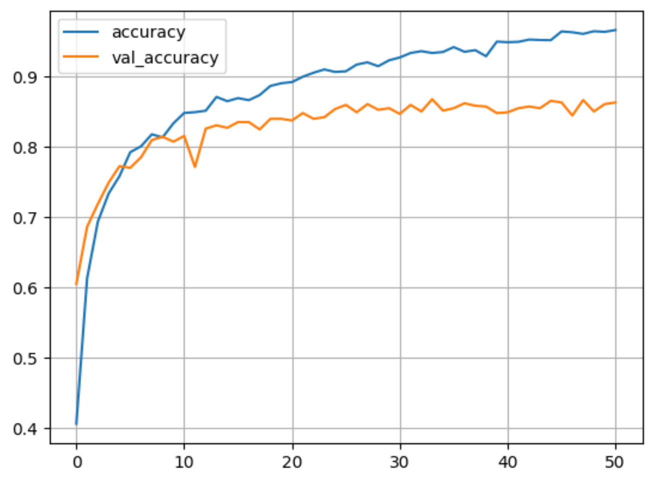
val_accuracy: 0.8628

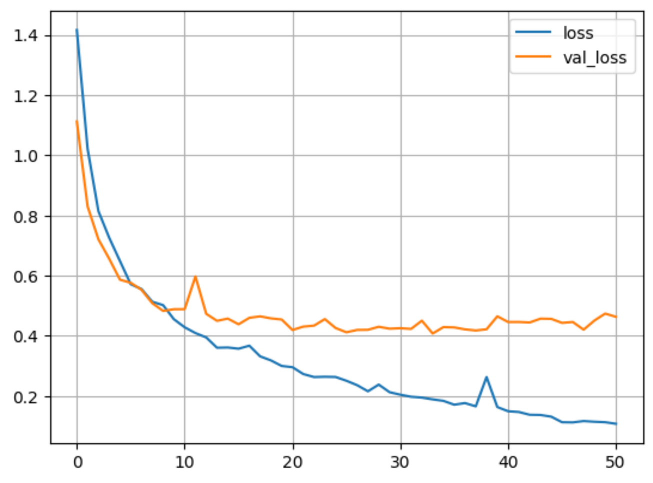
Tuned layer :

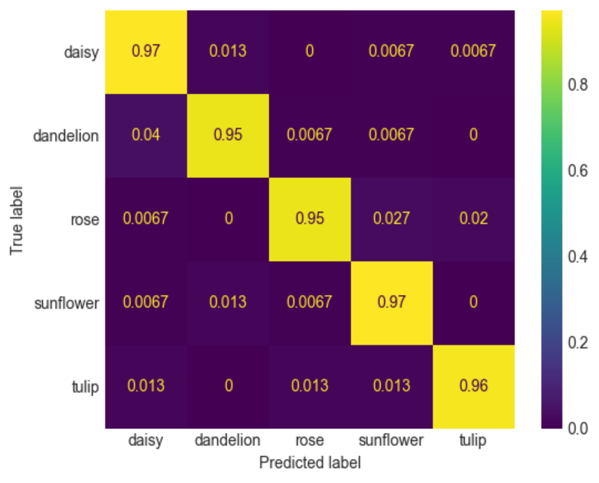
'block5_conv1'

'block5_conv2'

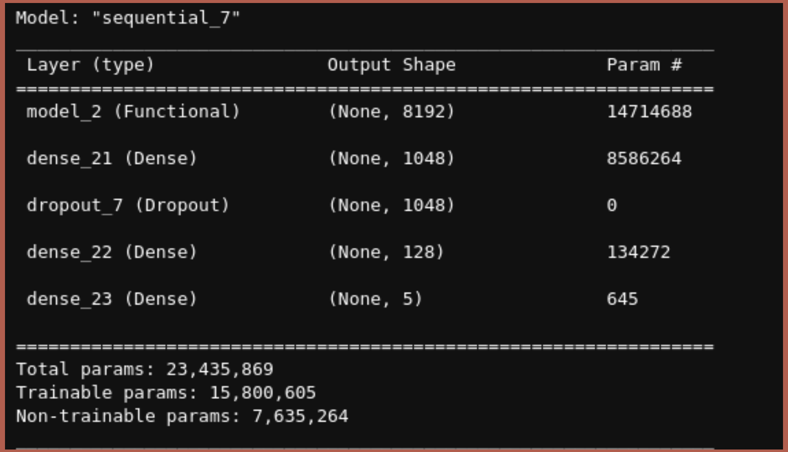
'block5_conv3'







True label \ Predicted label	daisy	dandelion	rose	sunflower	tulip
daisy	0.97	0.013	0	0.0067	0.0067
dandelion	0.04	0.95	0.0067	0.0067	0
rose	0.0067	0	0.95	0.027	0.02
sunflower	0.0067	0.013	0.0067	0.97	0
tulip	0.013	0	0.013	0.013	0.96



```
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 model_2 (Functional)        (None, 8192)              14714688

 dense_21 (Dense)            (None, 1048)              8586264

 dropout_7 (Dropout)         (None, 1048)              0

 dense_22 (Dense)            (None, 128)               134272

 dense_23 (Dense)            (None, 5)                 645

=================================================================
Total params: 23,435,869
Trainable params: 15,800,605
Non-trainable params: 7,635,264
_________________________________________________________________
```

Image with Predicted Labels

Google's Inception Convolutional Neural Network (InceptionV3)

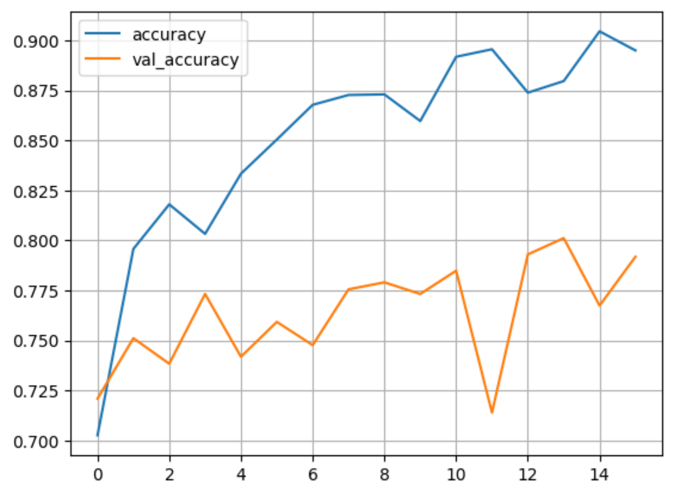
InceptionV3

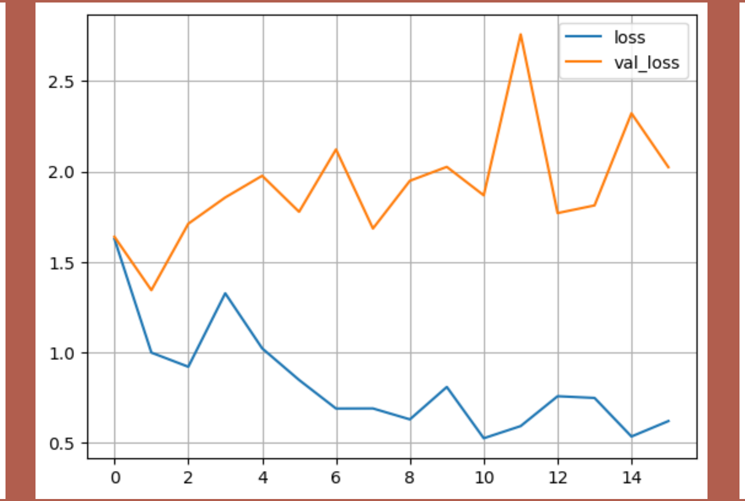
Optimizer :

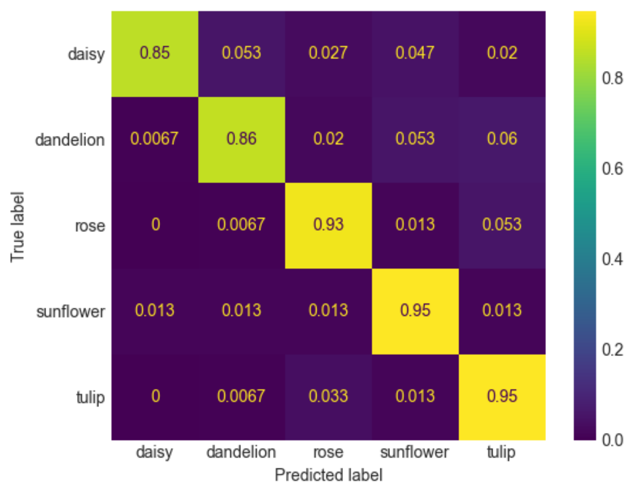
Adam

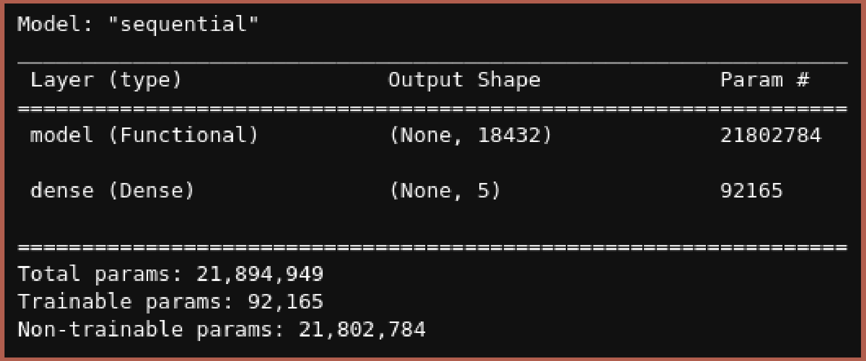
val_accuracy:

0.7919

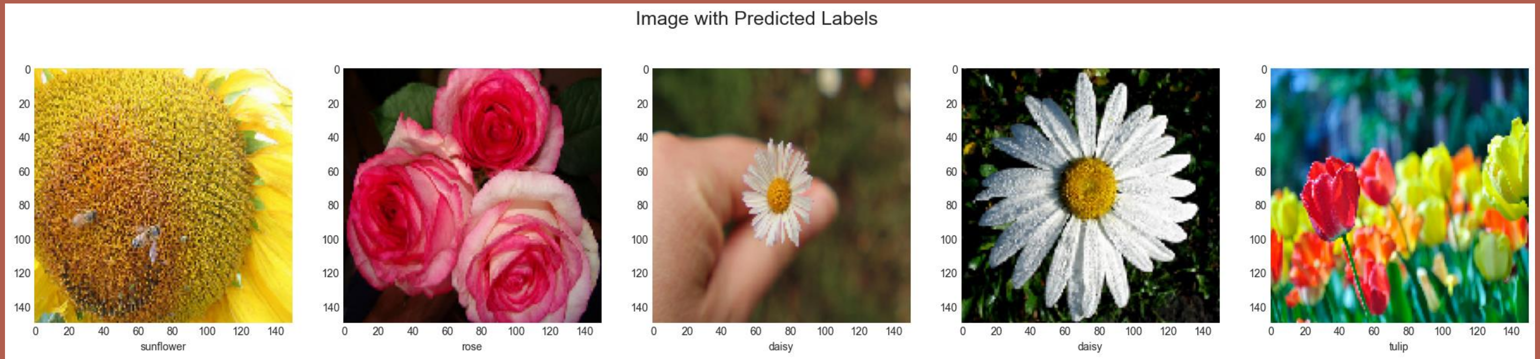








```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 model (Functional)          (None, 18432)             21802784

 dense (Dense)               (None, 5)                 92165

=================================================================
Total params: 21,894,949
Trainable params: 92,165
Non-trainable params: 21,802,784
```



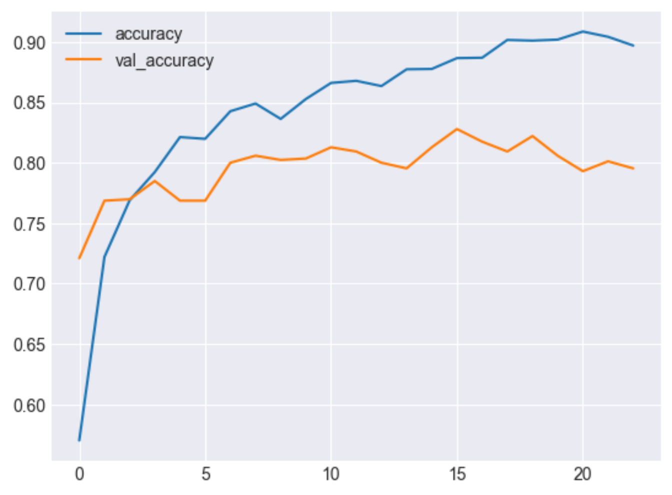
InceptionV3 with additional dense layer

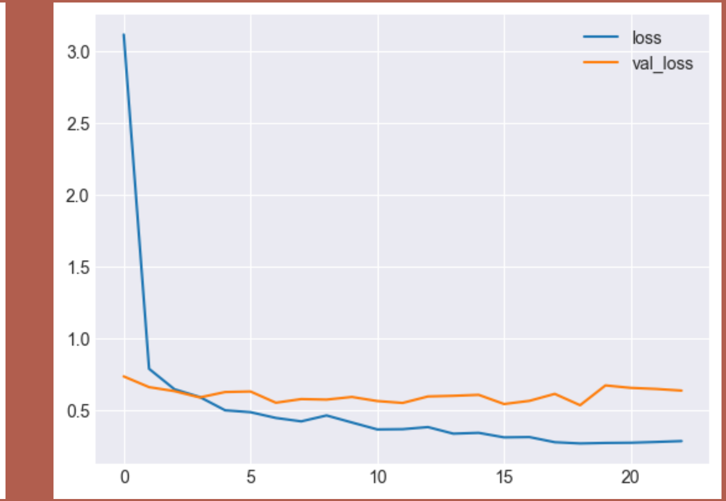
Optimizer :

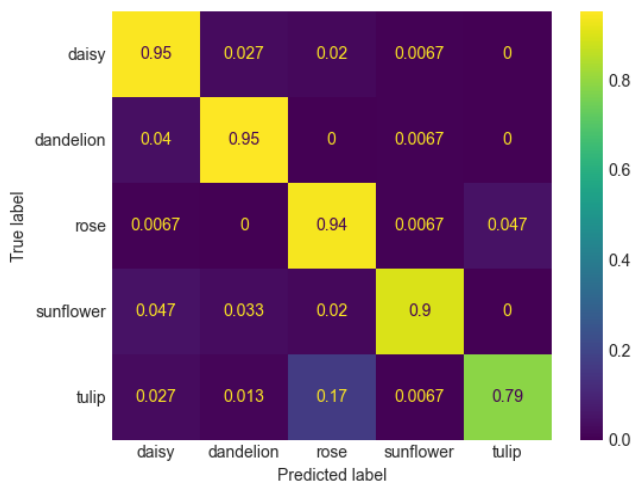
Adam

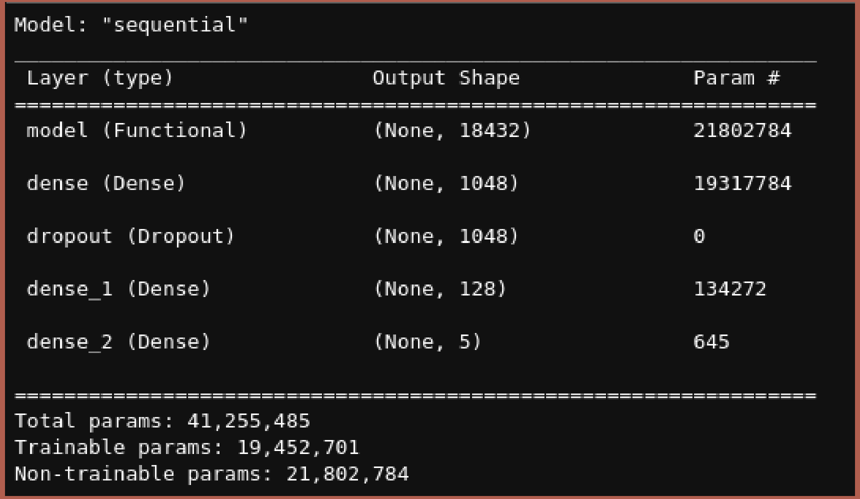
val_accuracy:

0.7953

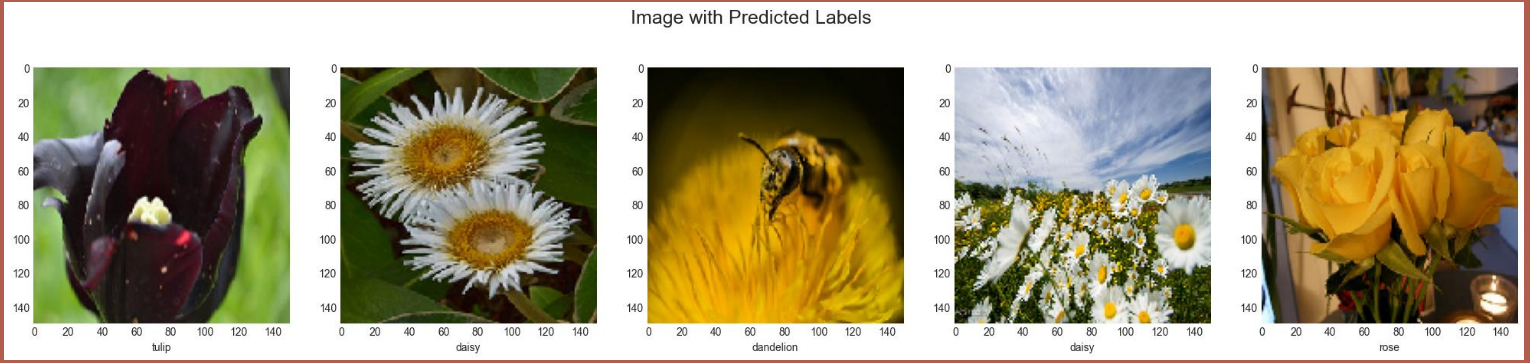








```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 model (Functional)          (None, 18432)             21802784

 dense (Dense)               (None, 1048)              19317784

 dropout (Dropout)           (None, 1048)              0

 dense_1 (Dense)             (None, 128)               134272

 dense_2 (Dense)             (None, 5)                 645

=================================================================
Total params: 41,255,485
Trainable params: 19,452,701
Non-trainable params: 21,802,784
```



InceptionV3 with Tuning

Optimizer :

SGD (lr = 0.001)

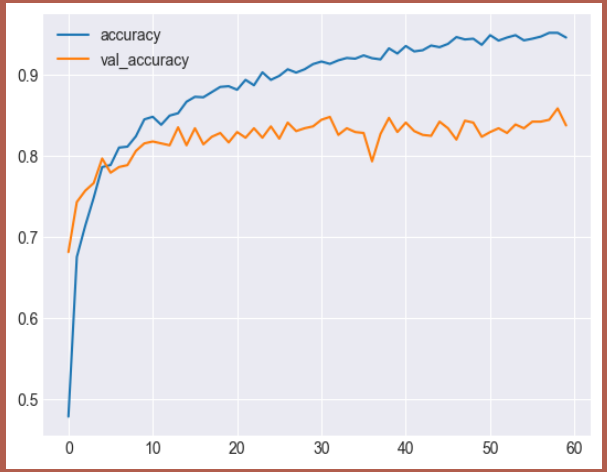
val_accuracy: 0.8372

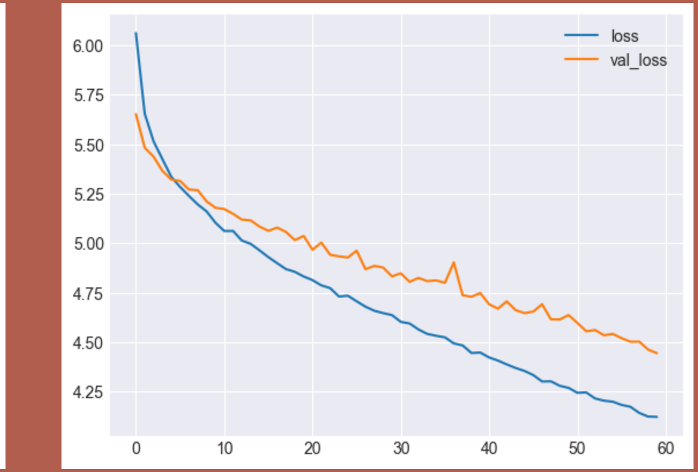
Tuned layer :

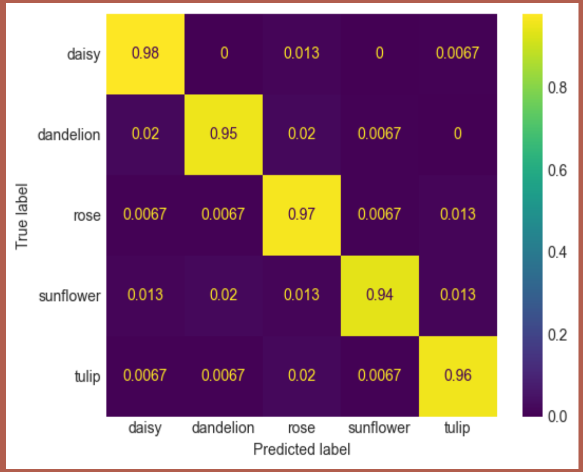
'conv2d_89',

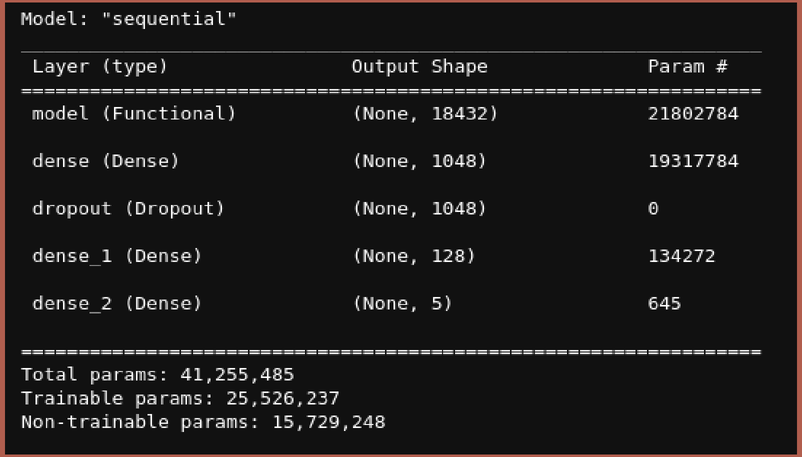
'conv2d_86',

'conv2d_85'









```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 model (Functional)          (None, 18432)             21802784

 dense (Dense)               (None, 1048)              19317784

 dropout (Dropout)           (None, 1048)              0

 dense_1 (Dense)             (None, 128)               134272

 dense_2 (Dense)             (None, 5)                 645

=================================================================
Total params: 41,255,485
Trainable params: 25,526,237
Non-trainable params: 15,729,248
```

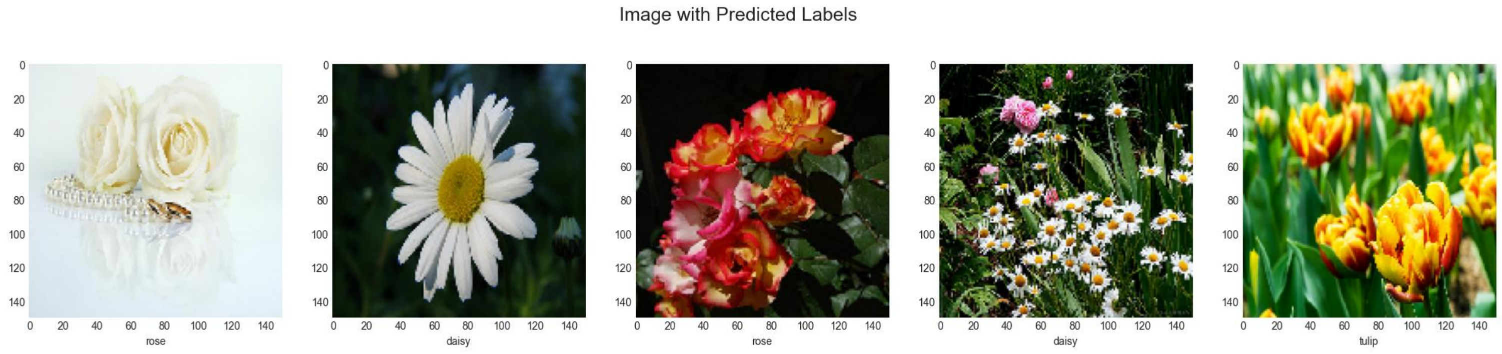


Image with Predicted Labels

Residual neural network （ResNet152V2）

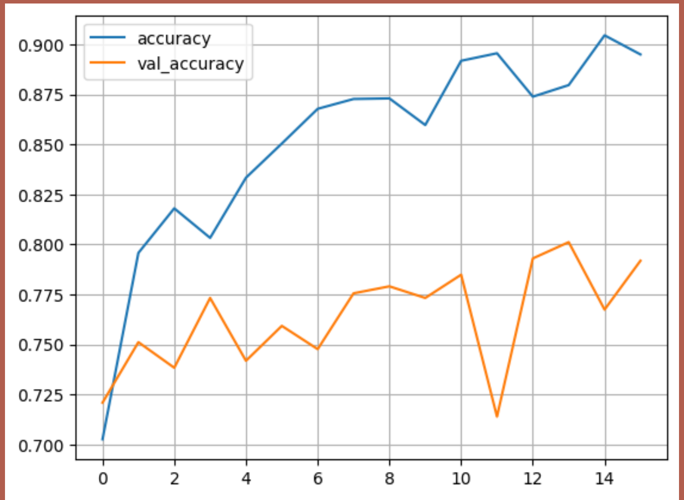
ResNet152V2

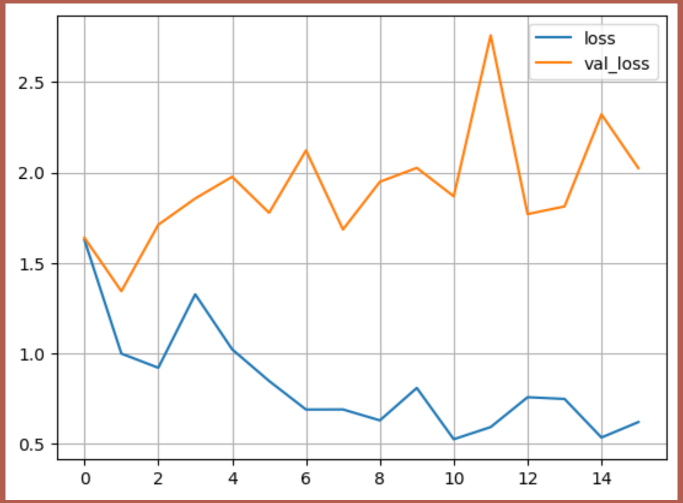
Optimizer :

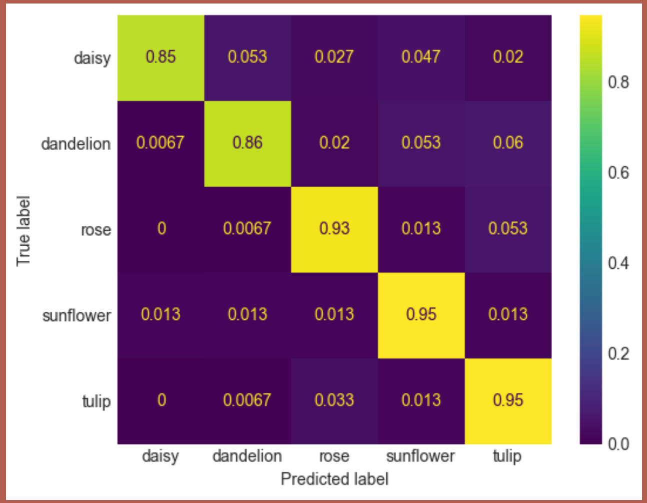
Adam

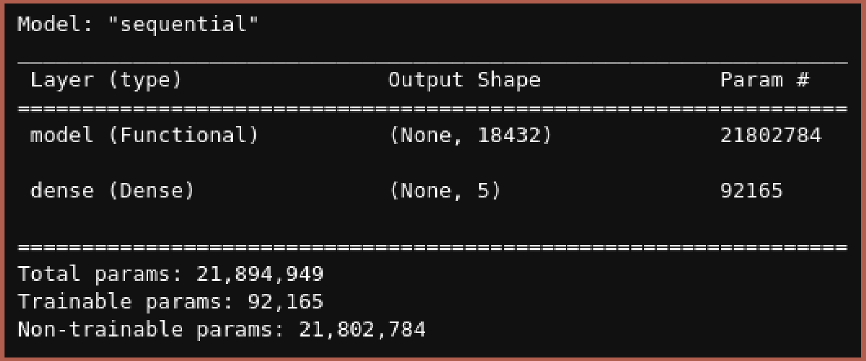
val_accuracy:

0.7919

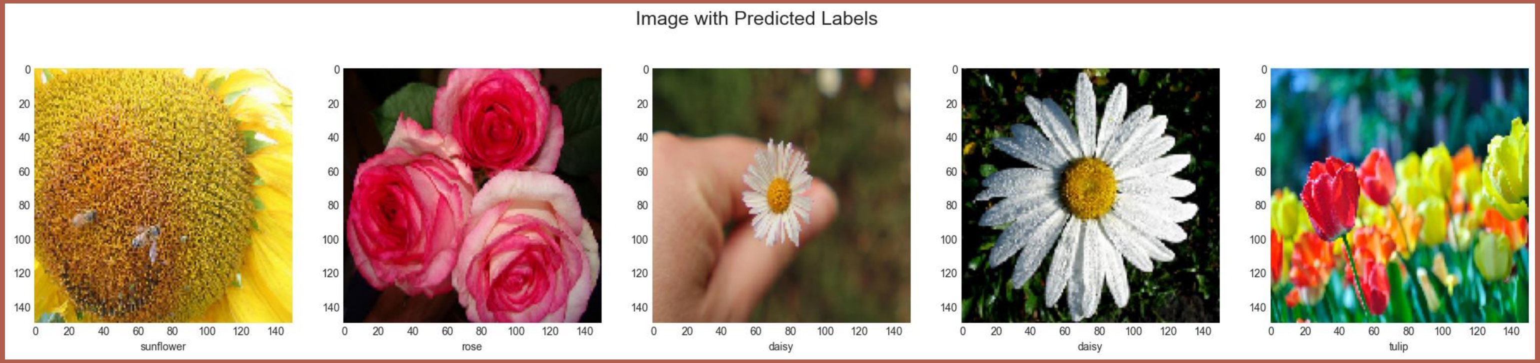








```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 model (Functional)          (None, 18432)             21802784

 dense (Dense)               (None, 5)                 92165

=================================================================
Total params: 21,894,949
Trainable params: 92,165
Non-trainable params: 21,802,784
```



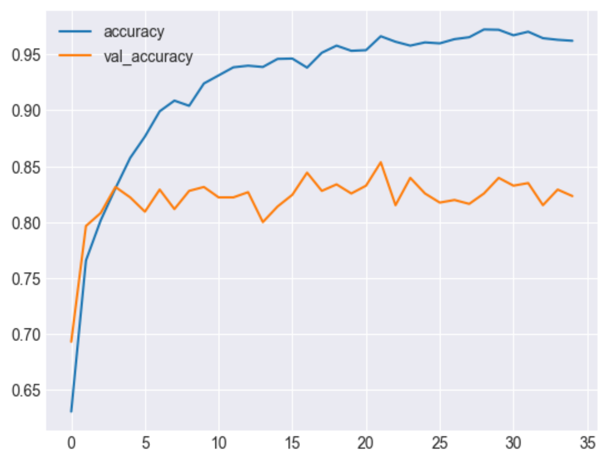
ResNet152V2 with additional dense layer

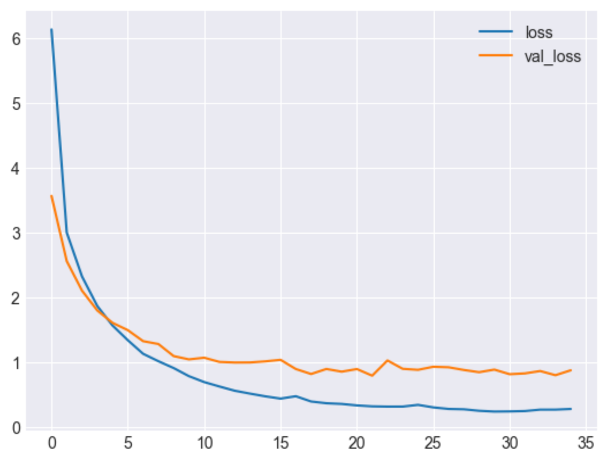
Optimizer :

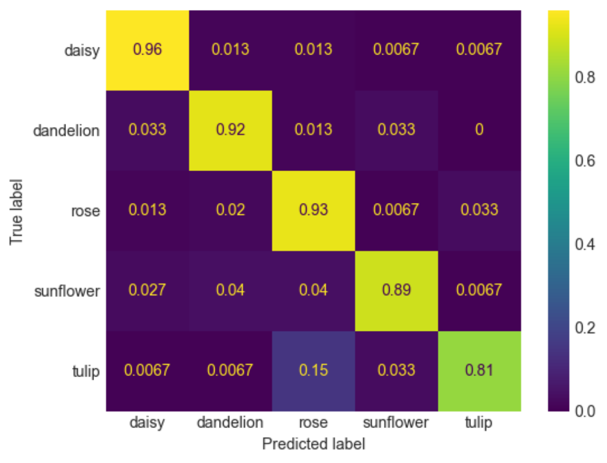
Adam

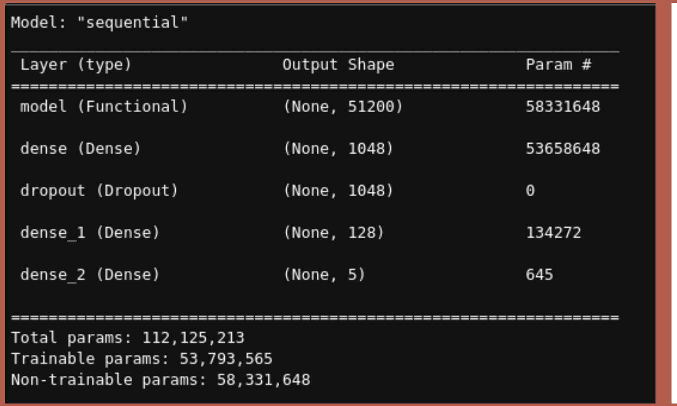
val_accuracy:

0.8233



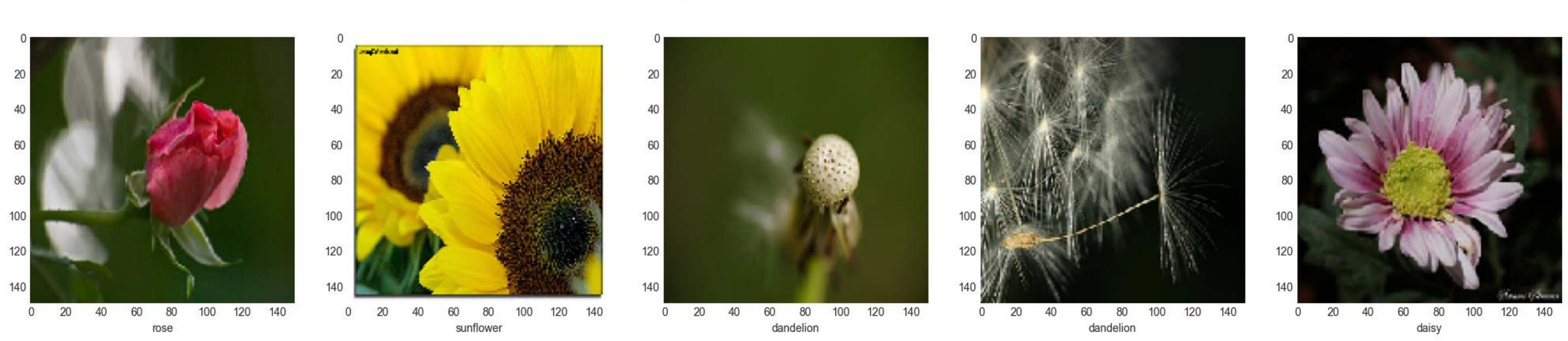






```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 model (Functional)          (None, 51200)             58331648

 dense (Dense)               (None, 1048)              53658648

 dropout (Dropout)           (None, 1048)              0

 dense_1 (Dense)             (None, 128)               134272

 dense_2 (Dense)             (None, 5)                 645

=================================================================
Total params: 112,125,213
Trainable params: 53,793,565
Non-trainable params: 58,331,648
```

Image with Predicted Labels



ResNet152V2 with Tuning

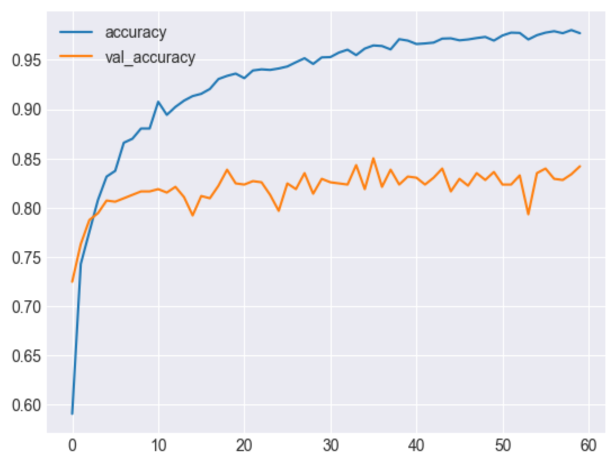
Optimizer :

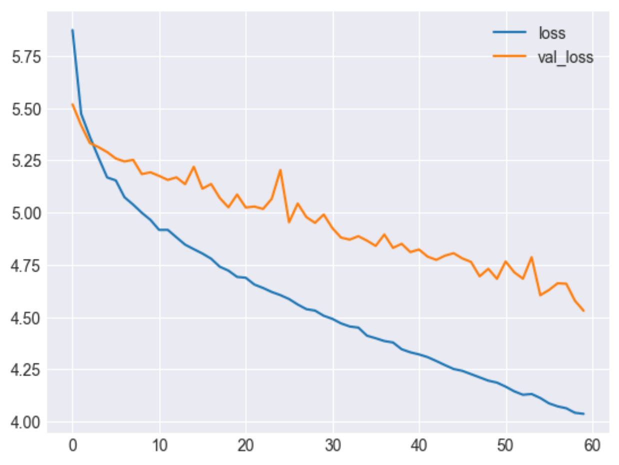
SGD (lr = 0.001)

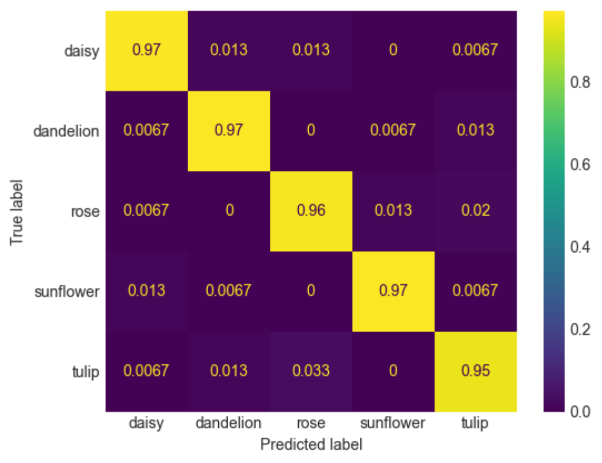
val_accuracy: 0.8419

Tuned layer :

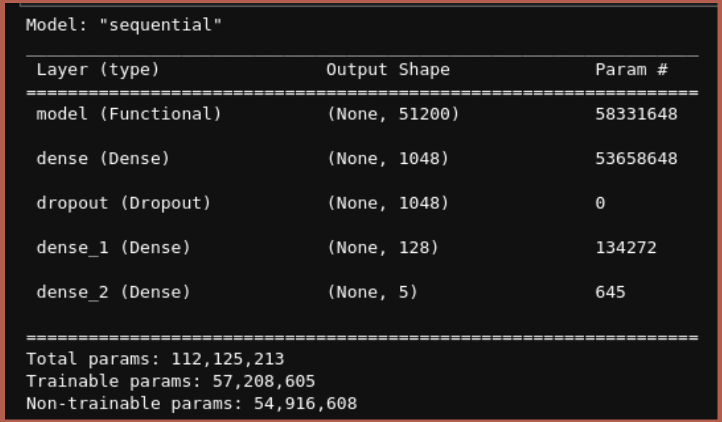
'conv5_block3_3_conv', 'conv5_block3_2_conv'







True label \ Predicted label	daisy	dandelion	rose	sunflower	tulip
daisy	0.97	0.013	0.013	0	0.0067
dandelion	0.0067	0.97	0	0.0067	0.013
rose	0.0067	0	0.96	0.013	0.02
sunflower	0.013	0.0067	0	0.97	0.0067
tulip	0.0067	0.013	0.033	0	0.95



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 model (Functional)          (None, 51200)             58331648

 dense (Dense)               (None, 1048)              53658648

 dropout (Dropout)           (None, 1048)              0

 dense_1 (Dense)             (None, 128)               134272

 dense_2 (Dense)             (None, 5)                 645

=================================================================
Total params: 112,125,213
Trainable params: 57,208,605
Non-trainable params: 54,916,608
```

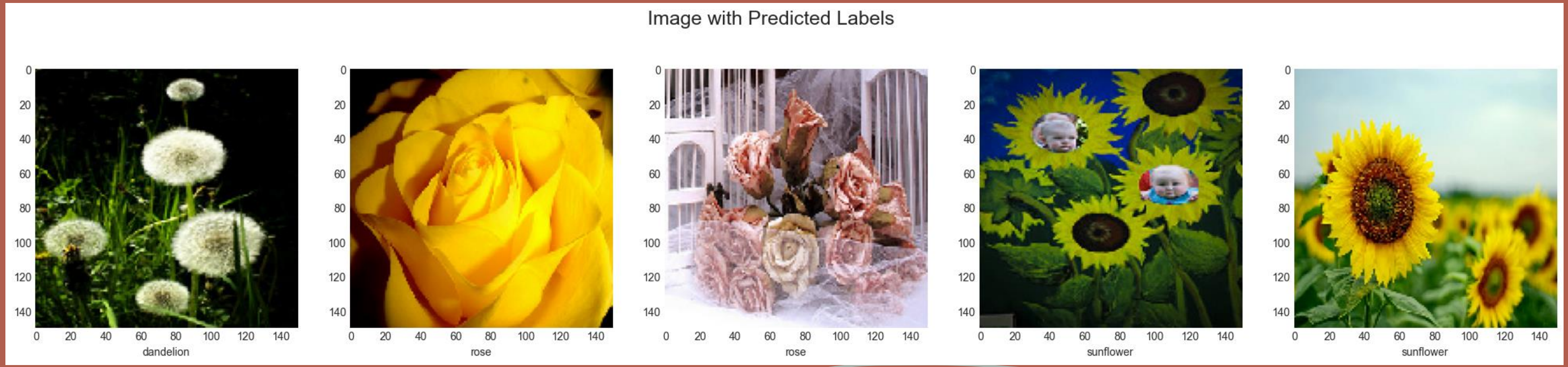
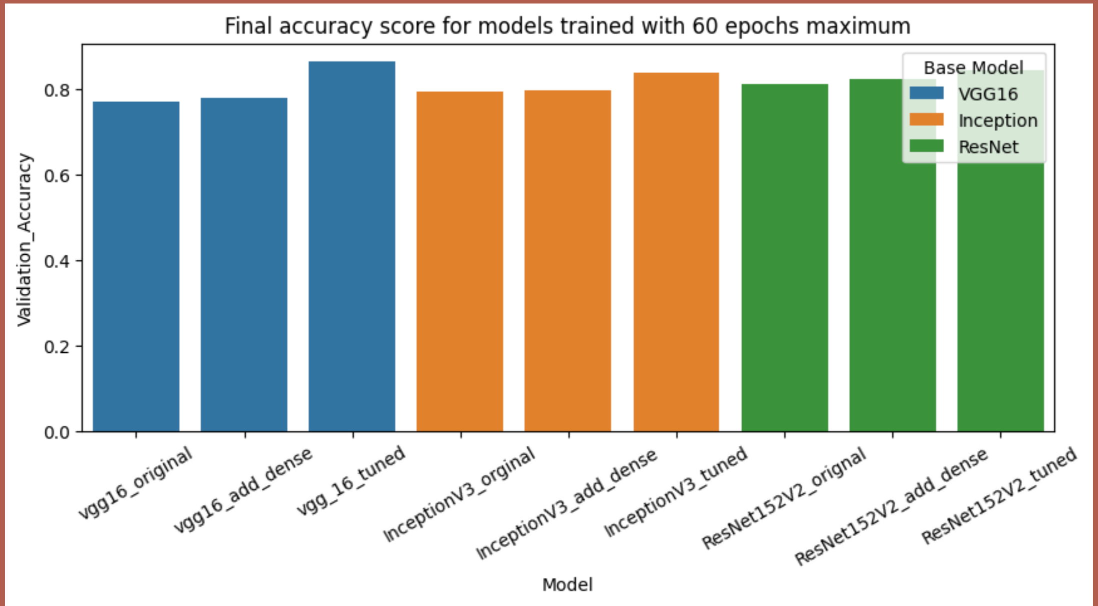
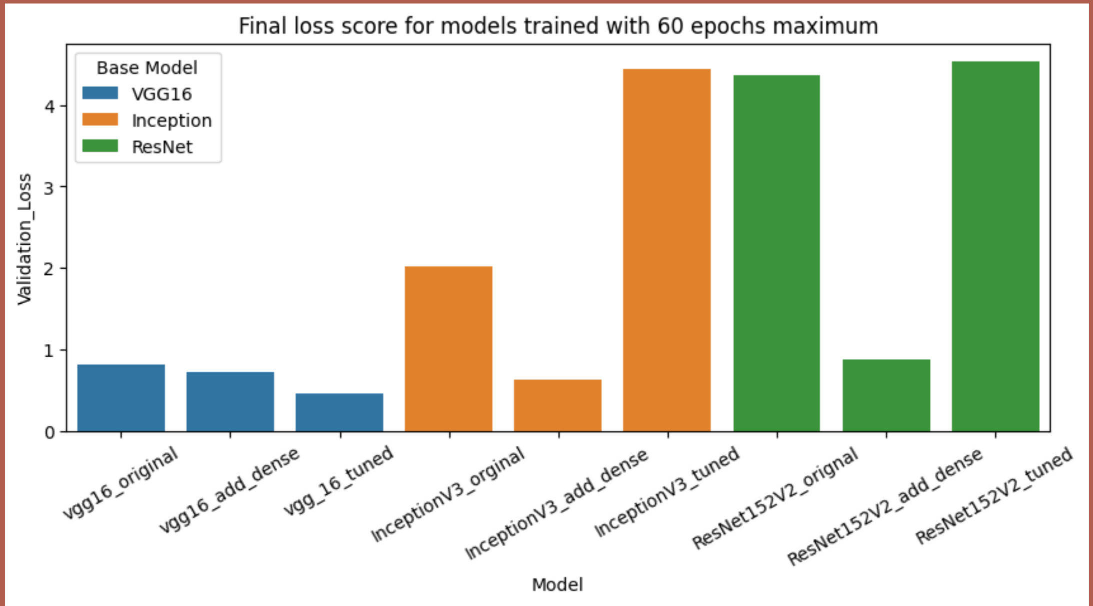


Image with Predicted Labels

Conclusion


Final accuracy score for models trained with 60 epochs maximum


Final loss score for models trained with 60 epochs maximum

- By adding more hidden layers or complexity, it always got slighter improve result.
- By tunning, later few layers, we get a huge improvement and got the best result.

Appendix

Github Link : https://github.com/thonenyangal/Image-Classification.git