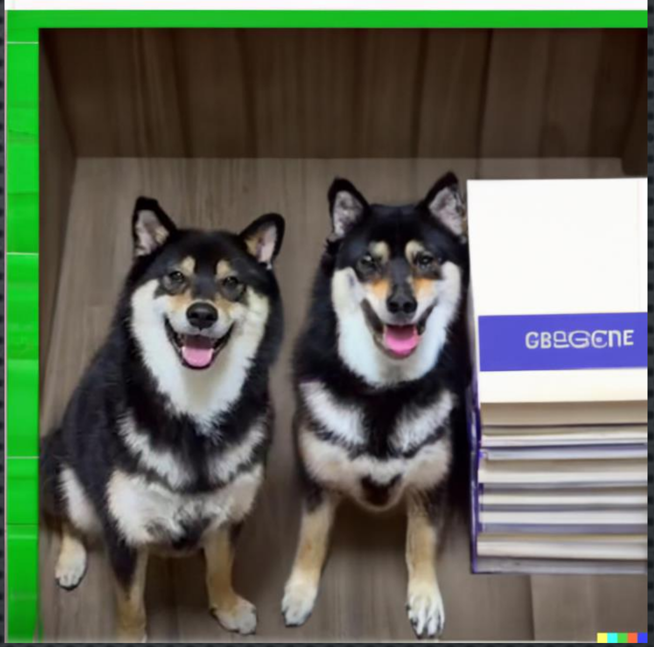
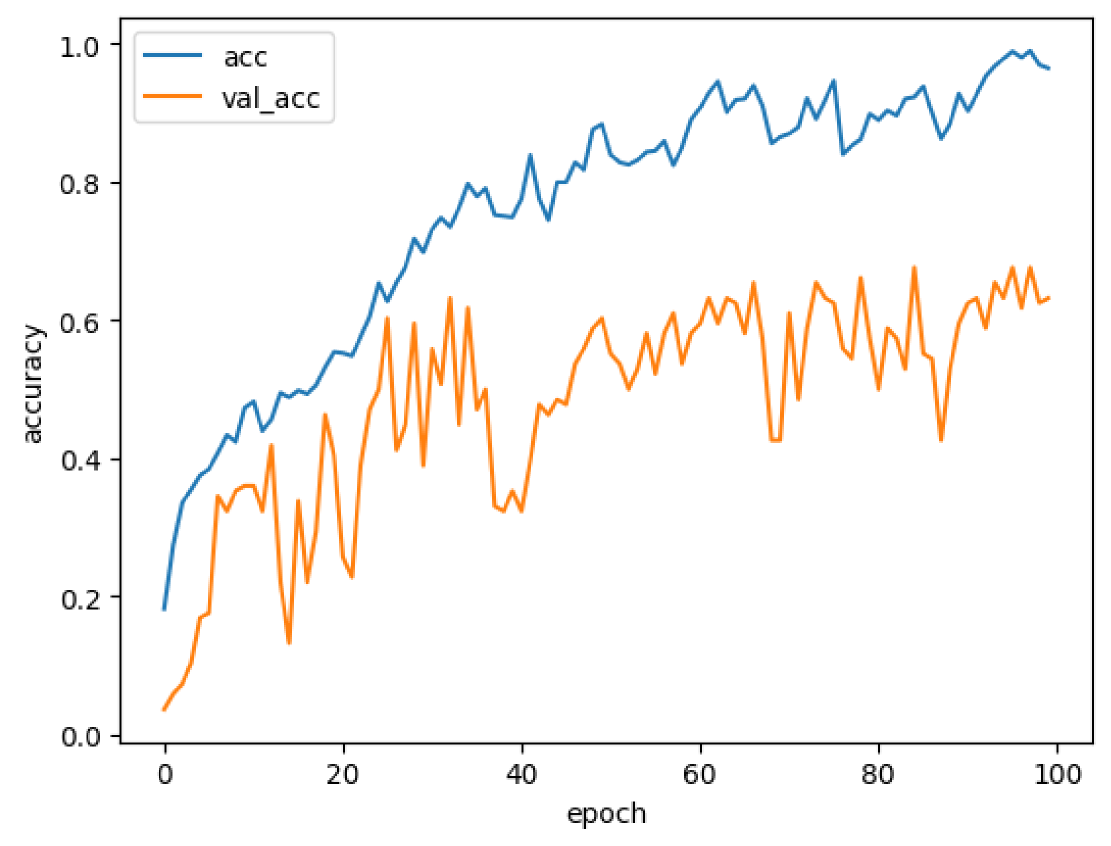
AI人工智慧—
進階CNN卷積神經網路



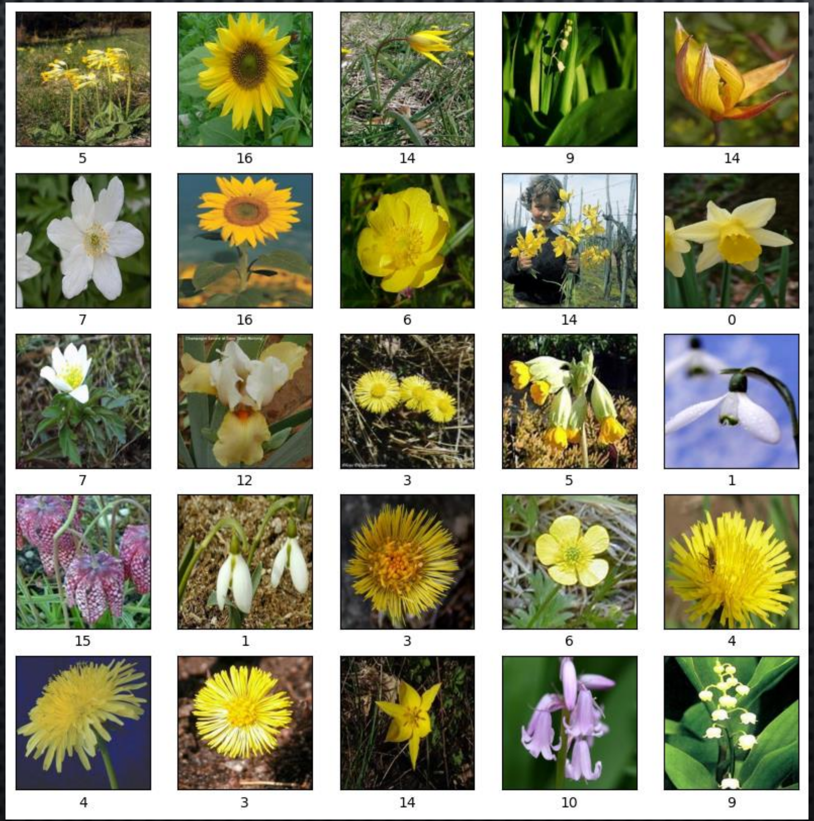
所謂進階CNN卷積神經網路learning by doing

von anwendeng

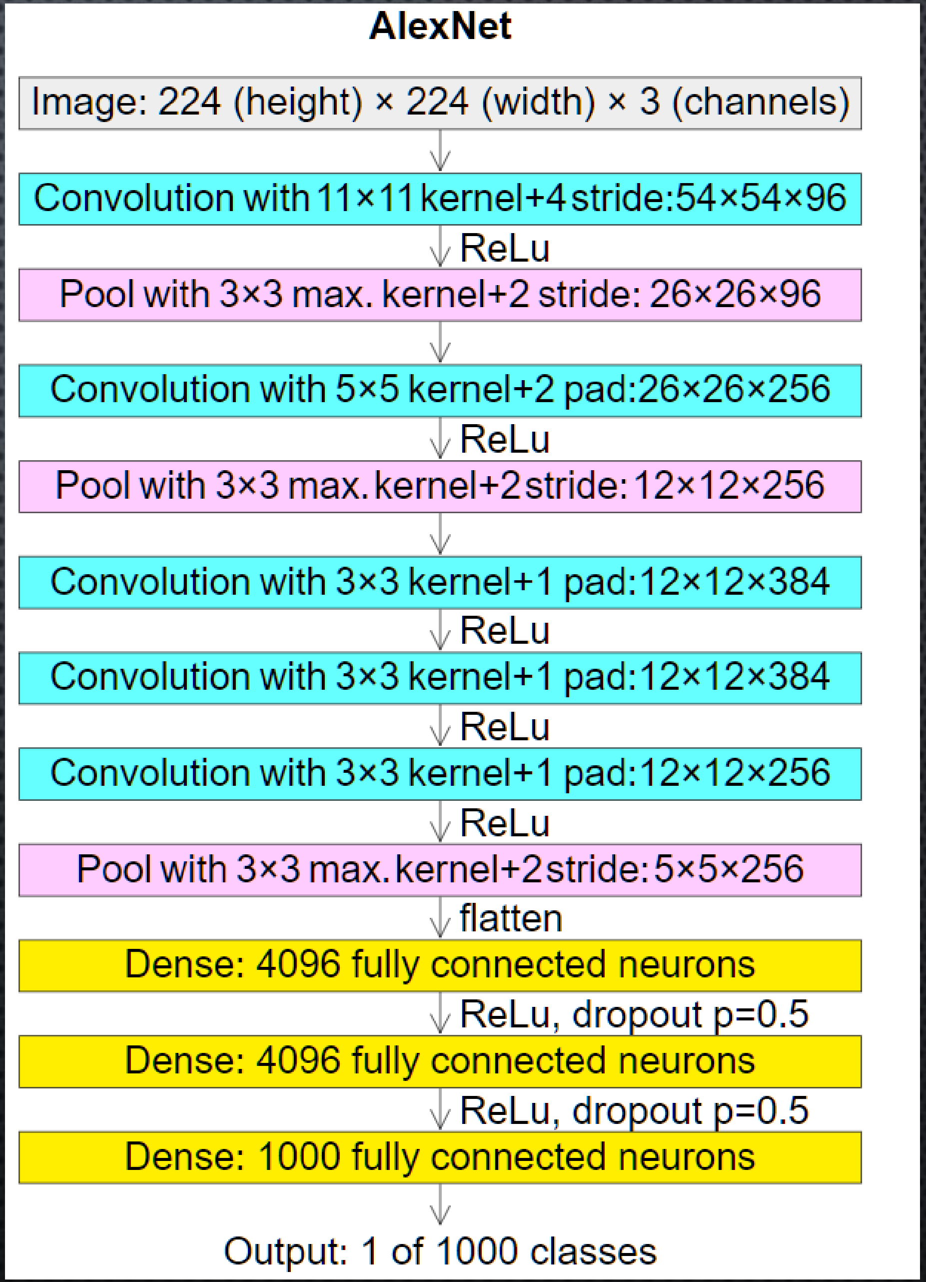
針對oxflower17的訓練成果，當然有進步的空間



```
predict:    [ 5 16 14  9 14  7 14  6 14  0  7 12  4  5  1  8  1  3  6  9  4  3 14 10
  9]
true value: [ 5 16 14  9 14  7 16  6 14  0  7 12  3  5  1 15  1  3  6  4  4  3 14 10
  9]
```



- By Cmglee - Own work, CC BY-SA 4.0, https://commons.wikimedia.org



from sklearn.model_selection import train_test_split

```
# 使用 train_test_split 函數將數據集分割為訓練集和測試集
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_s
```

Colab AI

您也可以使用其他套件將數據集分割為訓練集和測試集。例如，您可以使用 `numpy` 套件中的 `split` 函數。

```
import numpy as np

# 將數據集分割為訓練集和測試集
X_train, X_test, Y_train, Y_test = np.split(data, [i
```

Use code with caution

其中:

- `data` 是數據集
- `X_train` 是訓練集的特徵集
- `X_test` 是測試集的特徵集

您也可以使用 `pandas` 套件中的 `train_test_split` 函數。

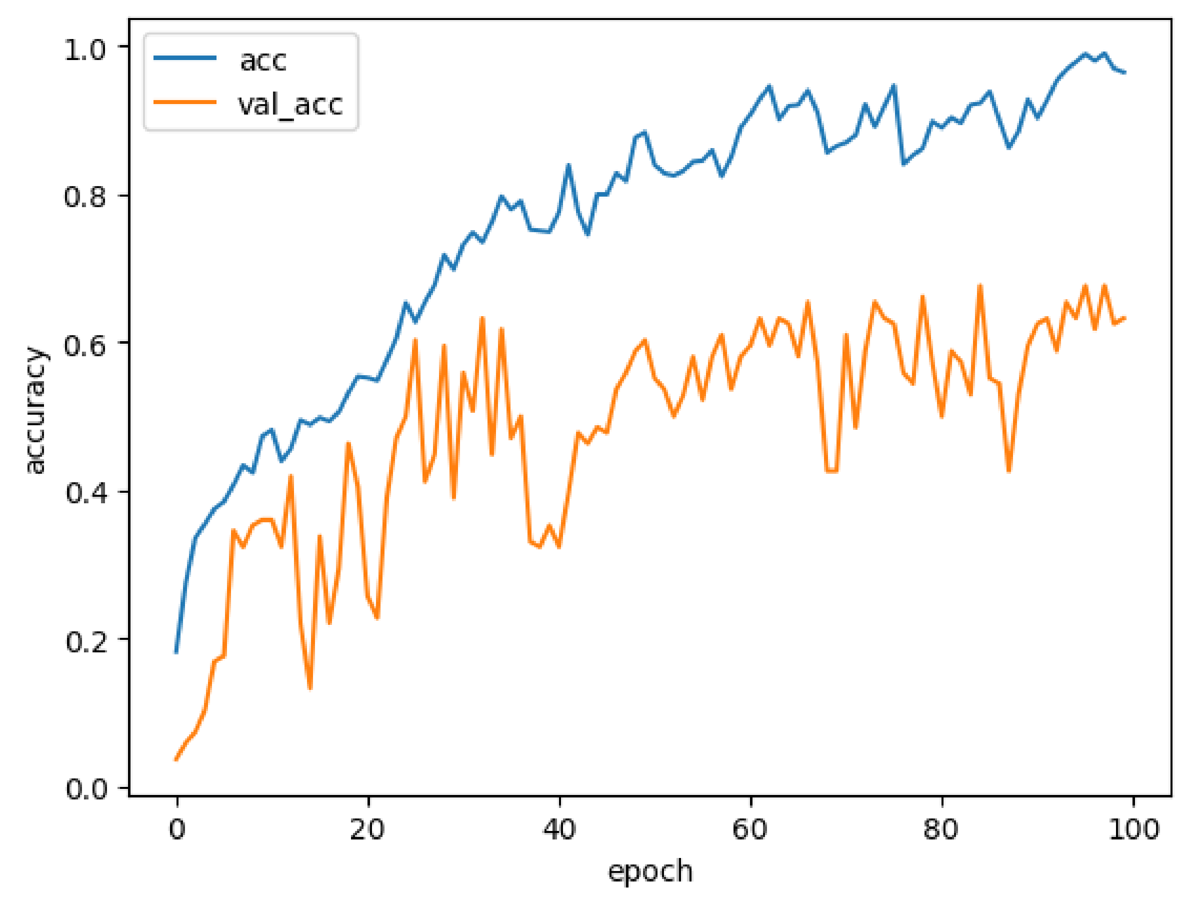
```
import pandas as pd

# 將數據集分割為訓練集和測試集
X_train, X_test, Y_train, Y_test = pd.train_test_split(data,
```

from keras.preprocessing.image import ImageDataGenerator

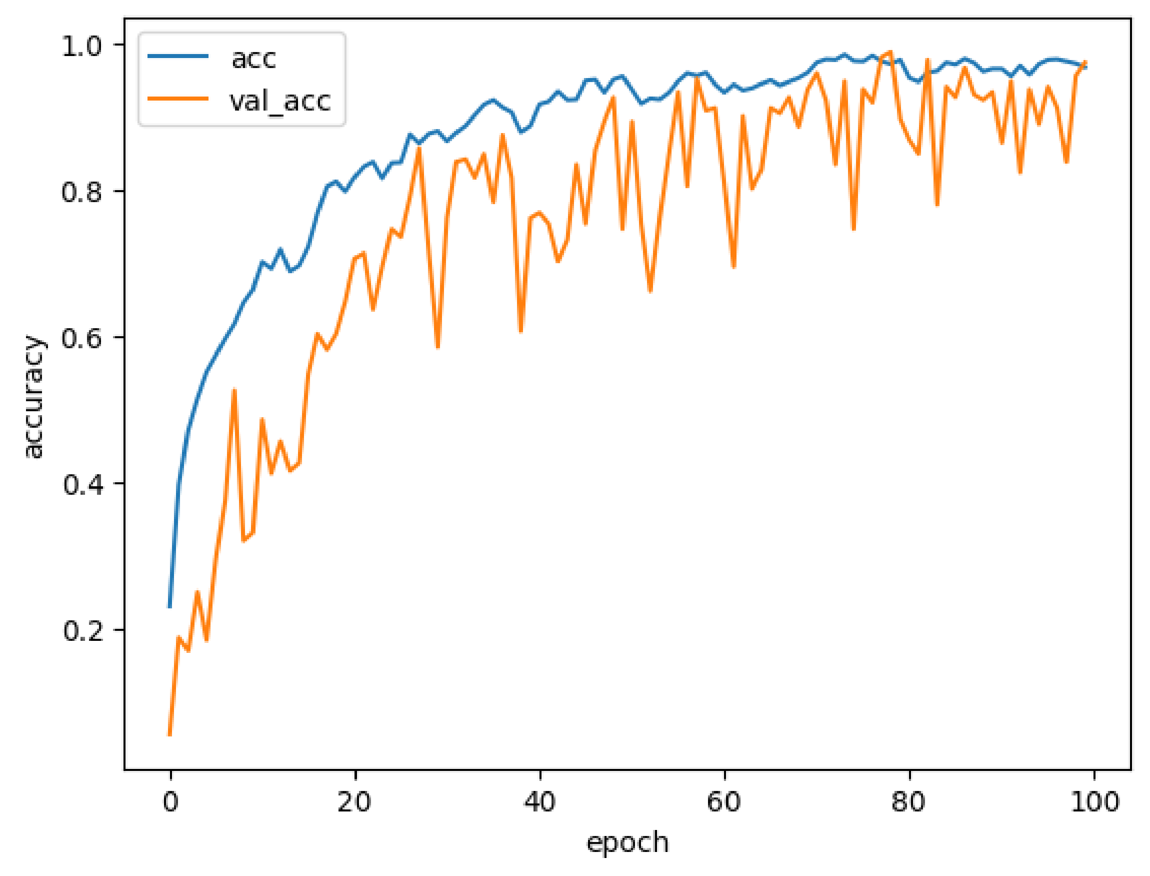
```
[ ]  from keras.preprocessing.image import ImageDataGenerator
```

```
# 使用 ImageDataGenerator 進行數據擴增
datagen = ImageDataGenerator(
        horizontal_flip=True,    # 左右鏡射
        rotation_range=30,    # 旋轉30度
    #   zca_whitening=False,    # 不進行 ZCA 白化處理
    #   featurewise_center=True    # 進行特徵均值中心化
)
```

V1 vs V2



acc
val_acc
accuracy
epoch
1.0
0.8
0.6
0.4
0.2
0.0
0
20
40
60
80
100



acc
val_acc
accuracy
epoch
1.0
0.8
0.6
0.4
0.2
0
20
40
60
80
100

Keras Applications

Keras Applications are deep learning models that are made available alongside pre-trained weights. These models can be used for prediction, feature extraction, and fine-tuning.

Weights are downloaded automatically when instantiating a model. They are stored at `~/.keras/models/`.

Upon instantiation, the models will be built according to the image data format set in your Keras configuration file at `~/.keras/keras.json`. For instance, if you have set `image_data_format=channels_last`, then any model loaded from this repository will get built according to the TensorFlow data format convention, "Height-Width-Depth".

Available models

Model	Size (MB)	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth	Time (ms) per inference step (CPU)	Time (ms) per inference step (GPU)
Xception	88	79.0%	94.5%	22.9M	81	109.4	8.1
VGG16	528	71.3%	90.1%	138.4M	16	69.5	4.2
VGG19	549	71.3%	90.0%	143.7M	19	84.8	4.4
ResNet50	98	74.9%	92.1%	25.6M	107	58.2	4.6
ResNet50V2	98	76.0%	93.0%	25.6M	103	45.6	4.4

- VGGNet使用3x3的filer size，的filer size，以及把深度加深，論文中提到總共有11~19的深度，其中以Vgg-16、Vgg-19效果最好。VGGNet在2014年ILSVRC的分類比賽中拿到亞軍

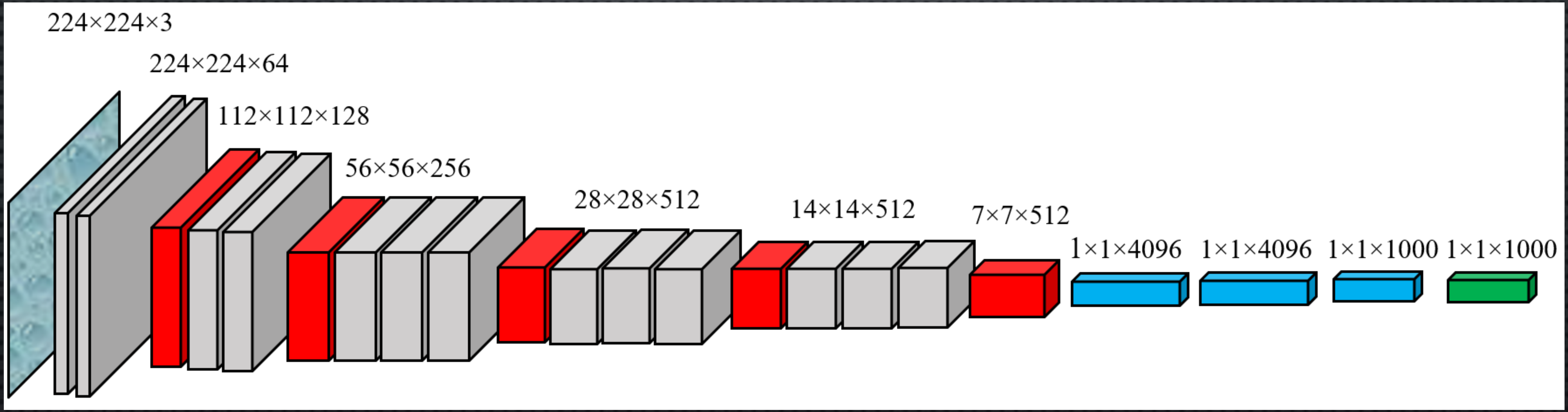
VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Karen Simonyan[*] **& Andrew Zisserman**[+]
Visual Geometry Group, Department of Engineering Science, University of Oxford
{karen,az}@robots.ox.ac.uk

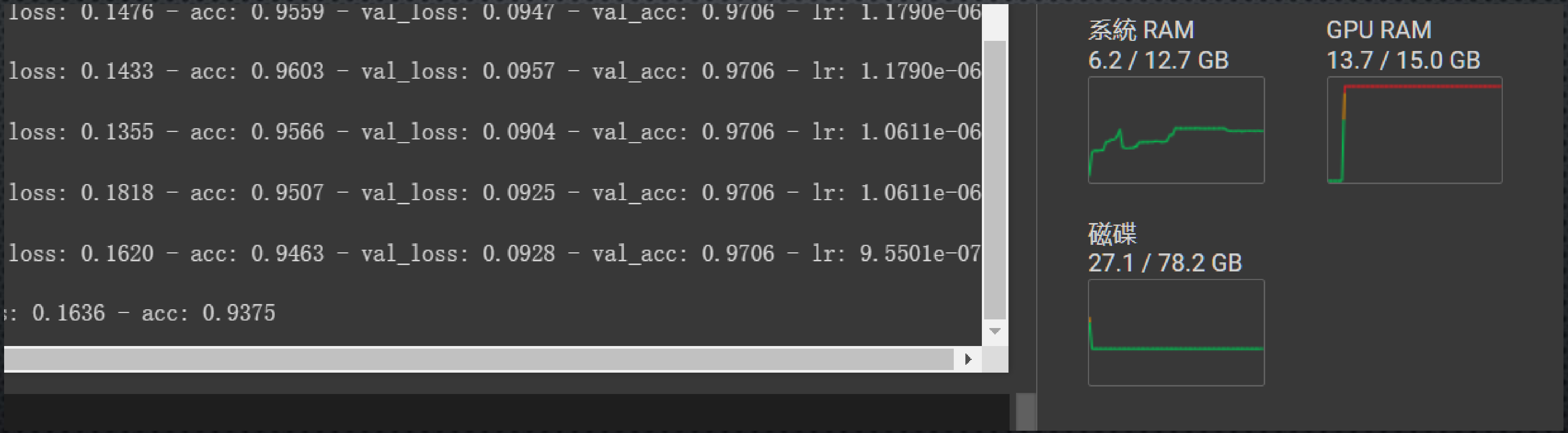
ABSTRACT

In this work we investigate the effect of the convolutional network depth on its accuracy in the large-scale image recognition setting. Our main contribution is a thorough evaluation of networks of increasing depth using an architecture with very small (3×3) convolution filters, which shows that a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16–19 weight layers. These findings were the basis of our ImageNet Challenge 2014 submission, where our team secured the first and the second places in the localisation and classification tracks respectively. We also show that our representations generalise well to other datasets, where they achieve state-of-the-art results. We have made our two best-performing ConvNet models publicly available to facili-

https://en.wikipedia.org/wiki/File:VGG_neural_network.png



CNN模型使用仿VGG



https://keras.io/api/callbacks/reduce_lr_on_plateau/

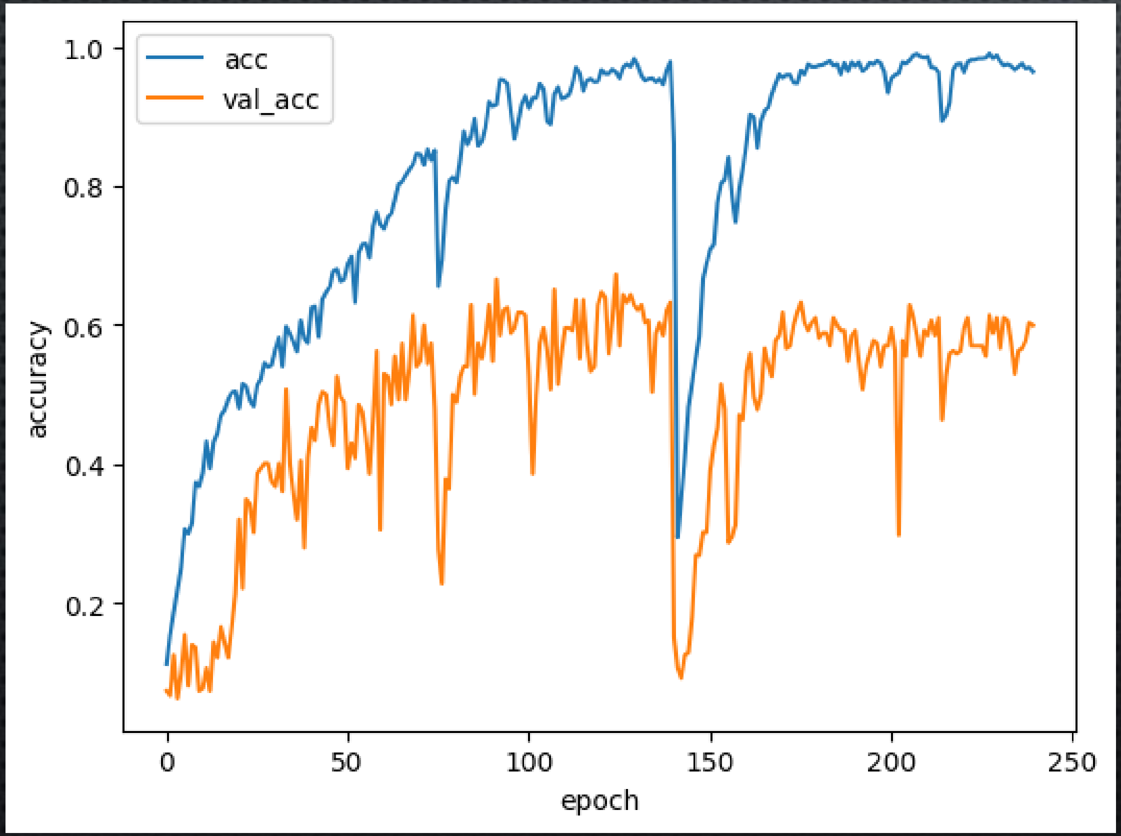
Reduce learning rate when a metric has stopped improving.

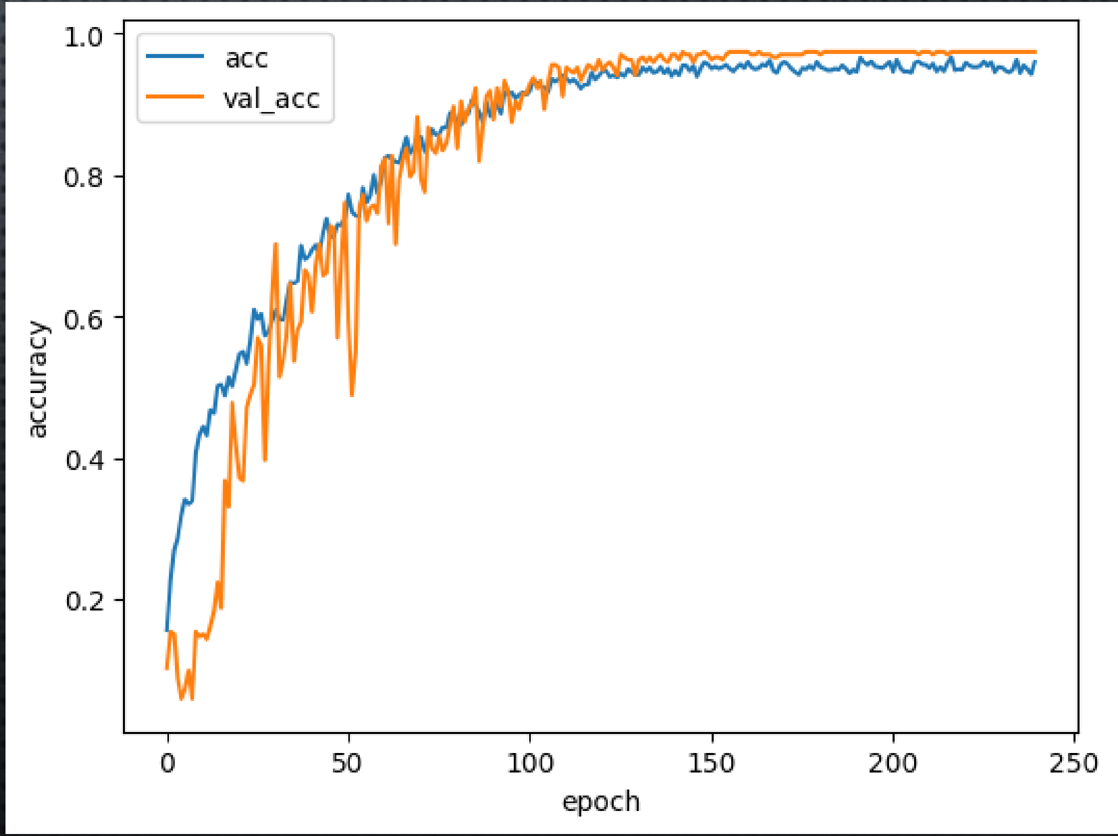
Models often benefit from reducing the learning rate by a factor of 2-10 once learning stagnates. This callback monitors a quantity and if no improvement is seen for a 'patience' number of epochs, the learning rate is reduced.

Example

```
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.2,
                              patience=5, min_lr=0.001)
model.fit(x_train, y_train, callbacks=[reduce_lr])
```

仿VGG V1 vs V2


acc
val_acc
accuracy
epoch
1.0
0.8
0.6
0.4
0.2
0
50
100
150
200
250


acc
val_acc
accuracy
epoch
1.0
0.8
0.6
0.4
0.2
0
50
100
150
200
250

軟體實作

von anwendeng

感謝觀賞

Herzlichen Dank für die Aufmerksamkeit

von anwendeng