File Edit View Insert Cell Kernel Navigate Widgets Help Trusted Python 3 (ipykernel)

Run Markdown

1 Listing 5.4. Copying images to training, validation, and test directories

In [ ]:
```
import os, shutil#處理檔案的套件

original_dataset_dir = 'C:/Users/Huang/Downloads/kaggle_original_data'

##創建三個檔案
base_dir = 'C:/Users/Huang/Downloads/cats_and_dogs_small'
os.mkdir(base_dir)
```

In [ ]:
```
train_dir = os.path.join(base_dir, 'train')
os.mkdir(train_dir)
validation_dir = os.path.join(base_dir, 'validation')
os.mkdir(validation_dir)
test_dir = os.path.join(base_dir, 'test')
os.mkdir(test_dir)

train_cats_dir = os.path.join(train_dir, 'cats')
os.mkdir(train_cats_dir)

train_dogs_dir = os.path.join(train_dir, 'dogs')
os.mkdir(train_dogs_dir)

validation_cats_dir = os.path.join(validation_dir, 'cats')
os.mkdir(validation_cats_dir)

validation_dogs_dir = os.path.join(validation_dir, 'dogs')
os.mkdir(validation_dogs_dir)

test_cats_dir = os.path.join(test_dir, 'cats')
os.mkdir(test_cats_dir)

test_dogs_dir = os.path.join(test_dir, 'dogs')
os.mkdir(test_dogs_dir)
```

In [10]:
```
fnames = ['cat.{}.jpg'.format(i) for i in range(1000)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(train_cats_dir, fname)
    shutil.copyfile(src, dst)

fnames = ['cat.{}.jpg'.format(i) for i in range(1000, 1500)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(validation_cats_dir, fname)
    shutil.copyfile(src, dst)

fnames = ['cat.{}.jpg'.format(i) for i in range(1500, 2000)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(test_cats_dir, fname)
    shutil.copyfile(src, dst)
########################貓狗大戰#########################
fnames = ['dog.{}.jpg'.format(i) for i in range(1000)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(train_dogs_dir, fname)
    shutil.copyfile(src, dst)
fnames = ['dog.{}.jpg'.format(i) for i in range(1000, 1500)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(validation_dogs_dir, fname)
    shutil.copyfile(src, dst)

fnames = ['dog.{}.jpg'.format(i) for i in range(1500, 2000)]
for fname in fnames:
```

```
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(test_dogs_dir, fname)
    shutil.copyfile(src, dst)
```

In [11]:

```
print('total training cat images:', len(os.listdir(train_cats_dir)))
print('total training dog images:', len(os.listdir(train_dogs_dir)))
print('total validation cat images:', len(os.listdir(validation_cats_dir)))
print('total validation dog images:', len(os.listdir(validation_dogs_dir)))
print('total test cat images:', len(os.listdir(test_cats_dir)))
print('total test dog images:', len(os.listdir(test_dogs_dir)))
```

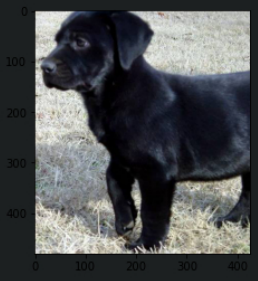
```
total training cat images: 1000
total training dog images: 1000
total validation cat images: 500
total validation dog images: 500
total test cat images: 500
total test dog images: 500
```

In [25]:

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from sklearn.model_selection import train_test_split
```

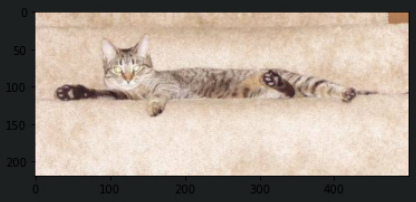
In [32]:

```
# display dog image
img = mpimg.imread('C:/Users/Huang/Downloads/kaggle_original_data/dog.4086.jpg')
imgplt = plt.imshow(img)
plt.show()
```



In [34]:

```
# display cat image
img = mpimg.imread('C:/Users/Huang/Downloads/kaggle_original_data/cat.5003.jpg')
imgplt = plt.imshow(img)
plt.show()
```



2 Listing 5.5. Instantiating a small convnet for dogs vs. cats classification

In [13]:

```
from keras import layers
from keras import models

model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu',
                        input_shape=(150, 150, 3)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
```

```
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))#因為只有兩類非貓及狗
```

In [14]:

```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 148, 148, 32)      896

 max_pooling2d_4 (MaxPooling  (None, 74, 74, 32)       0
 2D)

 conv2d_5 (Conv2D)           (None, 72, 72, 64)        18496

 max_pooling2d_5 (MaxPooling  (None, 36, 36, 64)       0
 2D)

 conv2d_6 (Conv2D)           (None, 34, 34, 128)       73856

 max_pooling2d_6 (MaxPooling  (None, 17, 17, 128)      0
 2D)

 conv2d_7 (Conv2D)           (None, 15, 15, 128)       147584

 max_pooling2d_7 (MaxPooling  (None, 7, 7, 128)        0
 2D)

 flatten (Flatten)           (None, 6272)              0

 dense (Dense)               (None, 512)               3211776

 dense_1 (Dense)             (None, 1)                 513

=================================================================
Total params: 3,453,121
Trainable params: 3,453,121
Non-trainable params: 0
_________________________________________________________________
```

3 Listing 5.6. Configuring the model for training

In [15]:

```
from keras import optimizers

model.compile(loss='binary_crossentropy',
              optimizer=optimizers.RMSprop(lr=1e-4),
              metrics=['acc'])
```

```
C:\Users\Huang\anaconda3\lib\site-packages\keras\optimizers\optimizer_v2\rmsprop.py:143: UserWarning: The `lr` argument is deprecated, use `learning_rate` instead.
  super().__init__(name, **kwargs)
```

4 Listing 5.7. Using ImageDataGenerator to read images from directories

In [17]:

```
from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale=1./255)
test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
        train_dir,
        target_size=(150, 150),
        batch_size=20,
        class_mode='binary')

validation_generator = test_datagen.flow_from_directory(
        validation_dir,
        target_size=(150, 150),
        batch_size=20,
        class_mode='binary')
```

```
Found 2000 images belonging to 2 classes.
Found 1000 images belonging to 2 classes.
```

In [19]:

```
for data_batch, labels_batch in train_generator:
    print('data batch shape:', data_batch.shape)
```

```
        print('labels batch shape:', labels_batch.shape)
        break
```

```
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
```

In [20]:

```
history = model.fit_generator(
      train_generator,
      steps_per_epoch=100,
      epochs=30,
      validation_data=validation_generator,
      validation_steps=50)
```

```
C:\Users\Huang\AppData\Local\Temp\ipykernel_13228\3259228942.py:1: UserWarning: `Model.fit_generator` is deprecated and will be re
moved in a future version. Please use `Model.fit`, which supports generators.
  history = model.fit_generator(


Epoch 1/30
100/100 [==============================] - 105s 1s/step - loss: 0.6865 - acc: 0.5435 - val_loss: 0.6693 - val_acc: 0.5800
Epoch 2/30
100/100 [==============================] - 88s 881ms/step - loss: 0.6590 - acc: 0.6205 - val_loss: 0.6445 - val_acc: 0.6240
Epoch 3/30
100/100 [==============================] - 90s 905ms/step - loss: 0.6212 - acc: 0.6580 - val_loss: 0.6398 - val_acc: 0.6340
Epoch 4/30
100/100 [==============================] - 89s 891ms/step - loss: 0.5793 - acc: 0.6920 - val_loss: 0.5983 - val_acc: 0.6710
Epoch 5/30
100/100 [==============================] - 85s 848ms/step - loss: 0.5415 - acc: 0.7260 - val_loss: 0.6175 - val_acc: 0.6610
Epoch 6/30
100/100 [==============================] - 84s 840ms/step - loss: 0.5172 - acc: 0.7450 - val_loss: 0.5705 - val_acc: 0.6900
Epoch 7/30
100/100 [==============================] - 79s 786ms/step - loss: 0.4893 - acc: 0.7665 - val_loss: 0.5698 - val_acc: 0.7070
Epoch 8/30
100/100 [==============================] - 78s 780ms/step - loss: 0.4614 - acc: 0.7835 - val_loss: 0.5931 - val_acc: 0.6870
Epoch 9/30
100/100 [==============================] - 79s 792ms/step - loss: 0.4372 - acc: 0.8010 - val_loss: 0.5499 - val_acc: 0.7110
Epoch 10/30
100/100 [==============================] - 79s 791ms/step - loss: 0.4075 - acc: 0.8130 - val_loss: 0.5885 - val_acc: 0.7110
Epoch 11/30
100/100 [==============================] - 81s 809ms/step - loss: 0.3845 - acc: 0.8250 - val_loss: 0.5822 - val_acc: 0.7150
Epoch 12/30
100/100 [==============================] - 78s 783ms/step - loss: 0.3690 - acc: 0.8395 - val_loss: 0.5942 - val_acc: 0.7040
Epoch 13/30
100/100 [==============================] - 78s 781ms/step - loss: 0.3394 - acc: 0.8590 - val_loss: 0.5477 - val_acc: 0.7470
Epoch 14/30
100/100 [==============================] - 81s 809ms/step - loss: 0.3209 - acc: 0.8660 - val_loss: 0.5872 - val_acc: 0.7260
Epoch 15/30
100/100 [==============================] - 79s 794ms/step - loss: 0.2957 - acc: 0.8755 - val_loss: 0.6354 - val_acc: 0.7180
Epoch 16/30
100/100 [==============================] - 79s 786ms/step - loss: 0.2808 - acc: 0.8845 - val_loss: 0.6315 - val_acc: 0.7270
Epoch 17/30
100/100 [==============================] - 78s 781ms/step - loss: 0.2635 - acc: 0.8910 - val_loss: 0.6247 - val_acc: 0.7300
Epoch 18/30
100/100 [==============================] - 78s 783ms/step - loss: 0.2428 - acc: 0.9055 - val_loss: 0.6142 - val_acc: 0.7270
Epoch 19/30
100/100 [==============================] - 80s 802ms/step - loss: 0.2240 - acc: 0.9155 - val_loss: 0.6167 - val_acc: 0.7300
Epoch 20/30
100/100 [==============================] - 78s 779ms/step - loss: 0.2033 - acc: 0.9200 - val_loss: 0.6292 - val_acc: 0.7430
Epoch 21/30
100/100 [==============================] - 78s 780ms/step - loss: 0.1867 - acc: 0.9300 - val_loss: 0.6371 - val_acc: 0.7390
Epoch 22/30
100/100 [==============================] - 81s 807ms/step - loss: 0.1612 - acc: 0.9520 - val_loss: 0.6780 - val_acc: 0.7410
Epoch 23/30
100/100 [==============================] - 78s 783ms/step - loss: 0.1495 - acc: 0.9480 - val_loss: 0.6602 - val_acc: 0.7410
Epoch 24/30
100/100 [==============================] - 77s 769ms/step - loss: 0.1252 - acc: 0.9580 - val_loss: 0.8013 - val_acc: 0.7300
Epoch 25/30
100/100 [==============================] - 78s 781ms/step - loss: 0.1204 - acc: 0.9575 - val_loss: 1.1057 - val_acc: 0.6830
Epoch 26/30
100/100 [==============================] - 78s 781ms/step - loss: 0.1011 - acc: 0.9690 - val_loss: 0.8717 - val_acc: 0.7300
Epoch 27/30
100/100 [==============================] - 81s 807ms/step - loss: 0.0887 - acc: 0.9710 - val_loss: 0.9081 - val_acc: 0.7260
Epoch 28/30
100/100 [==============================] - 78s 780ms/step - loss: 0.0701 - acc: 0.9835 - val_loss: 0.9023 - val_acc: 0.7220
Epoch 29/30
100/100 [==============================] - 77s 772ms/step - loss: 0.0716 - acc: 0.9790 - val_loss: 0.9069 - val_acc: 0.7320
Epoch 30/30
100/100 [==============================] - 78s 781ms/step - loss: 0.0602 - acc: 0.9830 - val_loss: 0.9502 - val_acc: 0.7370
```

5 Listing 5.9. Saving the model

In [21]:

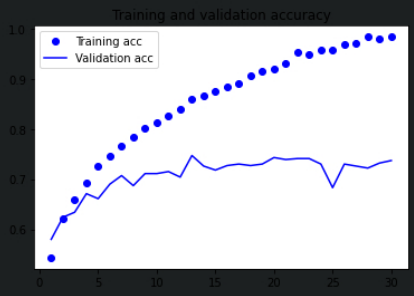
```
model.save('cats_and_dogs_small_1.h5')
```

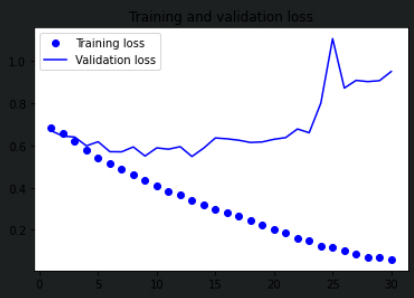
6 Listing 5.10. Displaying curves of loss and accuracy during training

In [22]:

```
import matplotlib.pyplot as plt
```

```
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(1, len(acc) + 1)

plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()
plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()

plt.show()
#12是最好
```





7 Listing 5.11. Setting up a data augmentation configuration via ImageDataGenerator

In [23]:

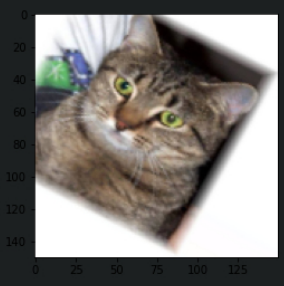
```
datagen = ImageDataGenerator(
      rotation_range=40,
      width_shift_range=0.2,
      height_shift_range=0.2,
      shear_range=0.2,
      zoom_range=0.2,
      horizontal_flip=True,
      fill_mode='nearest')
```

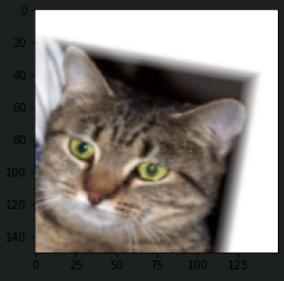
8 Listing 5.12. Displaying some randomly augmented training images

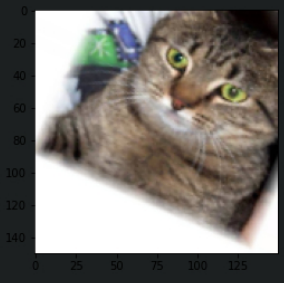
In [41]:

```
#from keras.preprocessing import image
import keras.utils as image

fnames = [os.path.join(train_cats_dir, fname) for fname in os.listdir(train_cats_dir)]

img_path = fnames[506]

img = image.load_img(img_path, target_size=(150, 150))

x = image.img_to_array(img)
```

```
x = x.reshape((1,) + x.shape)

i = 0
for batch in datagen.flow(x, batch_size=1):
    plt.figure(i)
    imgplot = plt.imshow(image.array_to_img(batch[0]))
    i += 1
    if i % 4 == 0:
        break

plt.show()
```







9 Listing 5.13. Defining a new convnet that includes dropout

In [42]:

```
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu',
                        input_shape=(150, 150, 3)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
```

```
model.add(layers.Flatten())
model.add(layers.Dropout(0.5))
model.add(layers.Dense(512, activation='relu'))

model.add(layers.Dense(1, activation='sigmoid'))

model.compile(loss='binary_crossentropy',
              optimizer=optimizers.RMSprop(lr=1e-4),
              metrics=['acc'])
```

```
C:\Users\Huang\anaconda3\lib\site-packages\keras\optimizers\optimizer_v2\rmsprop.py:143: UserWarning: The `lr` argument is deprecated, use `learning_rate` instead.
  super().__init__(name, **kwargs)
```

10 Listing 5.14. Training the convnet using data-augmentation generators

In [43]:

```
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,)

test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
        train_dir,
        target_size=(150, 150),
        batch_size=32,
        class_mode='binary')

validation_generator = test_datagen.flow_from_directory(
        validation_dir,
        target_size=(150, 150),
        batch_size=32,
        class_mode='binary')

history = model.fit_generator(
      train_generator,
      steps_per_epoch=100,
      epochs=100,
      validation_data=validation_generator,
      validation_steps=50)
```

...

model.save('cats_and_dogs_small_2.h5')