In [1]:

```
!pip install keras
import keras
```

```
Collecting keras
  Downloading https://files.pythonhosted.org/packages/ad/fd/6bfe87920d7f4fd475acd2
8500a42482b6b84479832bdc0fe9e589a60ceb/Keras-2.3.1-py2.py3-none-any.whl (377kB)
     |████████████████████████████████| 378kB 15.2
MB/s eta 0:00:01
Collecting pyyaml
  Downloading https://files.pythonhosted.org/packages/64/c2/b80047c7ac2478f9501676
c988a5411ed5572f35d1beff9cae07d321512c/PyYAML-5.3.1.tar.gz (269kB)
     |████████████████████████████████| 276kB 13.3
MB/s eta 0:00:01
Requirement already satisfied: keras-applications>=1.0.6 in /usr/local/lib/python
3.6/dist-packages (from keras) (1.0.8)
Requirement already satisfied: keras-preprocessing>=1.0.5 in /usr/local/lib/python
3.6/dist-packages (from keras) (1.1.0)
Requirement already satisfied: scipy>=0.14 in /usr/local/lib/python3.6/dist-packag
es (from keras) (1.4.1)
Requirement already satisfied: six>=1.9.0 in /usr/lib/python3/dist-packages (from
keras) (1.11.0)
Requirement already satisfied: numpy>=1.9.1 in /usr/local/lib/python3.6/dist-packa
ges (from keras) (1.17.2)
Requirement already satisfied: h5py in /usr/local/lib/python3.6/dist-packages (fro
m keras) (2.10.0)
Building wheels for collected packages: pyyaml
  Building wheel for pyyaml (setup.py) ... done
  Created wheel for pyyaml: filename=PyYAML-5.3.1-cp36-cp36m-linux_x86_64.whl size
=45919 sha256=8f74c1f324f9ebdd5155219d197ca7138053cfac59b41c0dae0ea91a58f84bc0
  Stored in directory: /root/.cache/pip/wheels/a7/c1/ea/cf5bd31012e735dc1dfea3131a
2d5eae7978b251083d6247bd
Successfully built pyyaml
Installing collected packages: pyyaml, keras
Successfully installed keras-2.3.1 pyyaml-5.3.1
WARNING: You are using pip version 19.3.1; however, version 20.0.2 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
```

```
Using TensorFlow backend.
```

In [2]:

```
import tensorflow as tf
tf.test.gpu_device_name()
```

Out[2]:

```
'/device:GPU:0'
```

In [3]:

```
!unzip -o -q BoatOrCar.zip
```

In [4]:

```
import numpy as np
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
```

In [6]:

```
from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255)
training_set = train_datagen.flow_from_directory('DataBoatOrCar/training',
                                                 target_size = (128, 128),
                                                 batch_size = 32,
                                                 class_mode = 'binary')

test_datagen = ImageDataGenerator(rescale = 1./255)
test_set = test_datagen.flow_from_directory('DataBoatOrCar/test',
                                            target_size = (128, 128),
                                            batch_size = 32,
                                            class_mode = 'binary')
from matplotlib import pyplot as plt
x,y = training_set.next()
for i in range(0,10):
    image = x[i]
    plt.imshow(image)
    plt.show()
```

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

<Figure size 640x480 with 1 Axes>

<Figure size 640x480 with 1 Axes>

<Figure size 640x480 with 1 Axes>

<Figure size 640x480 with 1 Axes>

<Figure size 640x480 with 1 Axes>

<Figure size 640x480 with 1 Axes>

<Figure size 640x480 with 1 Axes>

<Figure size 640x480 with 1 Axes>

<Figure size 640x480 with 1 Axes>

<Figure size 640x480 with 1 Axes>
```

In [9]:

```
classifier = Sequential()

classifier.add(Conv2D(32, (3, 3), input_shape = (128, 128, 3), activation = 'relu'))

classifier.add(MaxPooling2D(pool_size = (2, 2)))

classifier.add(Conv2D(32, (3, 3), activation = 'relu'))

classifier.add(MaxPooling2D(pool_size = (2, 2)))

classifier.add(Conv2D(32, (3, 3), activation = 'relu'))

classifier.add(Flatten())

classifier.add(Dense(units = 128, activation = 'relu'))
classifier.add(Dense(units = 128, activation = 'relu'))
classifier.add(Dense(units = 128, activation = 'relu'))

classifier.add(Dense(units = 1, activation = 'sigmoid'))

classifier.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])

classifier.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_4 (Conv2D)            (None, 126, 126, 32)      896
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 63, 63, 32)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 61, 61, 32)        9248
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 30, 30, 32)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 28, 28, 32)        9248
_________________________________________________________________
flatten_2 (Flatten)          (None, 25088)             0
_________________________________________________________________
dense_5 (Dense)              (None, 128)               3211392
_________________________________________________________________
dense_6 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_7 (Dense)              (None, 128)               16512
_________________________________________________________________
dense_8 (Dense)              (None, 1)                 129
=================================================================
Total params: 3,263,937
Trainable params: 3,263,937
Non-trainable params: 0
_________________________________________________________________
```

In [10]:

```
history = classifier.fit_generator(training_set,
                         steps_per_epoch = 2000,
                         epochs = 5,
                         validation_data = test_set,
                         validation_steps = 1000)

classifier.save("BoatOrCarModel.hdf5") # Hierarchical Data Format Fifth Version
print("Saved model to disk")

import pickle
filehandler = open('history.pkl', 'wb')
pickle.dump(history, filehandler)
```

```
Epoch 1/5
2000/2000 [==============================] - 1351s 676ms/step - loss: 0.0642 - acc
uracy: 0.9728 - val_loss: 1.1397 - val_accuracy: 0.8551
Epoch 2/5
2000/2000 [==============================] - 1348s 674ms/step - loss: 7.4446e-04 -
accuracy: 0.9993 - val_loss: 2.1193 - val_accuracy: 0.8557
Epoch 3/5
2000/2000 [==============================] - 1358s 679ms/step - loss: 0.0181 - acc
uracy: 0.9936 - val_loss: 0.3384 - val_accuracy: 0.8173
Epoch 4/5
2000/2000 [==============================] - 1343s 671ms/step - loss: 0.0038 - acc
uracy: 0.9986 - val_loss: 0.1763 - val_accuracy: 0.8210
Epoch 5/5
2000/2000 [==============================] - 1345s 673ms/step - loss: 7.1588e-04 -
accuracy: 0.9993 - val_loss: 3.0874 - val_accuracy: 0.8171
Saved model to disk
```

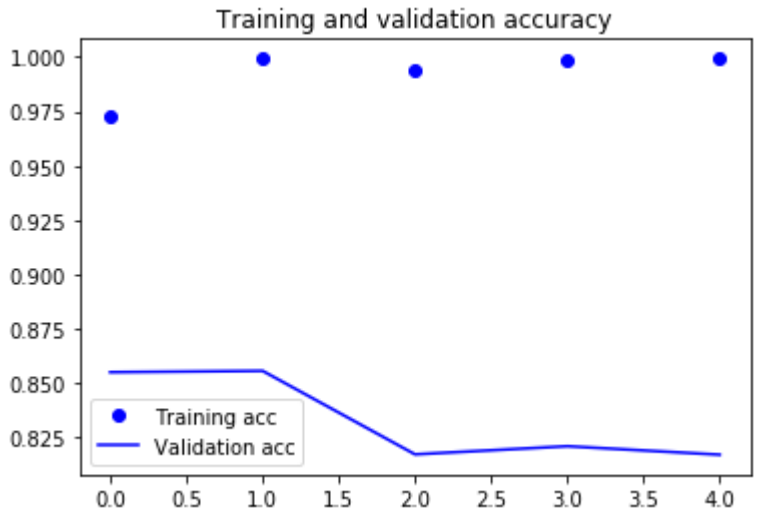
In [11]:

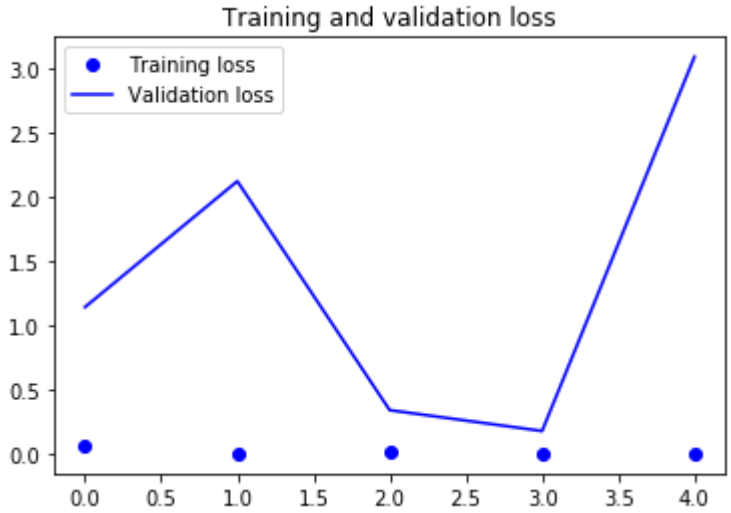
```
from keras.models import load_model
classifier = load_model('BoatOrCarModel.hdf5')

print(classifier.evaluate(training_set))

print(classifier.evaluate(test_set))

test_set.reset()
y_pred = classifier.predict_generator(test_set)
y_test = test_set.classes[test_set.index_array]
y_pred = ((y_pred>0.5)*1.0).flatten()
print(sum(y_pred==y_test)/y_pred.size)

!pip install sklearn
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
63/63 [==============================] - 33s 521ms/step
[2.1102204428302684e-09, 0.9994999766349792]
32/32 [==============================] - 16s 514ms/step
[8.795751571655273, 0.8169999718666077]
0.817
Collecting sklearn
  Downloading https://files.pythonhosted.org/packages/1e/7a/dbb3be0ce9bd5c8b7e3d87
328e79063f8b263b2b1bfa4774cb1147bfcd3f/sklearn-0.0.tar.gz
Collecting scikit-learn
  Downloading https://files.pythonhosted.org/packages/5e/d8/312e03adf4c78663e17d80
2fe2440072376fee46cada1404f1727ed77a32/scikit_learn-0.22.2.post1-cp36-cp36m-manyli
nux1_x86_64.whl (7.1MB)
     |████████████████████████████████████████████████████████████████| 7.1MB 11.8
MB/s eta 0:00:01
Requirement already satisfied: scipy>=0.17.0 in /usr/local/lib/python3.6/dist-pack
ages (from scikit-learn->sklearn) (1.4.1)
Collecting joblib>=0.11
  Downloading https://files.pythonhosted.org/packages/28/5c/cf6a2b65a321c4a209efcd
f64c2689efae2cb62661f8f6f4bb28547cf1bf/joblib-0.14.1-py2.py3-none-any.whl (294kB)
     |████████████████████████████████████████████████████████████████| 296kB 49.4
MB/s eta 0:00:01
Requirement already satisfied: numpy>=1.11.0 in /usr/local/lib/python3.6/dist-pack
ages (from scikit-learn->sklearn) (1.17.2)
Building wheels for collected packages: sklearn
  Building wheel for sklearn (setup.py) ... done
  Created wheel for sklearn: filename=sklearn-0.0-py2.py3-none-any.whl size=2397 s
ha256=5b8b1f97b52c34315f48a0f5cad9ada17b6d25feb5d19427b55f9f2d3412dd77
  Stored in directory: /root/.cache/pip/wheels/76/03/bb/589d421d27431bcd2c6da284d5
f2286c8e3b2ea3cf1594c074
Successfully built sklearn
Installing collected packages: joblib, scikit-learn, sklearn
Successfully installed joblib-0.14.1 scikit-learn-0.22.2.post1 sklearn-0.0
WARNING: You are using pip version 19.3.1; however, version 20.0.2 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
[[402  98]
 [ 85 415]]
```

In [12]:

```
import pickle
with open('history.pkl', 'rb') as pickle_file:
    content = pickle.load(pickle_file)

import matplotlib.pyplot as plt
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(len(acc))

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()
```



In [ ]:

```
```