Dog breed classifier

We selected 9 dog breeds for this project, and use 13599 pictures (about 1500 for each breed) for training and 1447 pitcures for testing.

In this project, our method is to use bottleneck featuers from VGG16 and train by only the top layers because it's computational efficient.

In [10]:

```
import keras
```

In [11]:

```
from keras import applications
from keras.preprocessing.image import ImageDataGenerator
from keras import optimizers
from keras.models import Sequential
from keras.layers import Dropout, Flatten, Dense
from keras.utils import np_utils
from glob import glob
import math
```

In [12]:

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [13]:

```
import keras.backend as K
K.clear_session()
```

We choose input pictures size as 150x150.

In [14]:

```
nrow = 150
ncol = 150
```

Load VGG16 as our base model.

In [15]:

```
input_shape = (nrow, ncol,3)
base_model = applications.VGG16(include_top=False, weights='imagenet',input_shap
e=input_shape)
```

Load datas

Rescale data values from 0 to 1, and set class_mode=None, shuffle=False.
We tried to set class_mode='categorical' and shuffle=True, but the accracy we got from that case wasn't as good as this one.

In [16]:

```
train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
batch_size = 32
train_data_dir = 'Images/train'
train_generator = train_datagen.flow_from_directory(
                        train_data_dir,
                        target_size=(nrow,ncol),
                        batch_size=batch_size,
                        class_mode=None,shuffle=False)


ntrain_samples = len(train_generator.filenames)
num_classes = len(train_generator.class_indices)
predict_size_train = int(math.ceil(ntrain_samples/batch_size))
```

```
Found 13599 images belonging to 9 classes.
```

In [17]:

```
test_data_dir = 'Images/test'
test_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
batch_size = 32
test_generator = test_datagen.flow_from_directory(
                        test_data_dir,
                        target_size=(nrow,ncol),
                        batch_size=batch_size,
                        class_mode=None,shuffle=False)
ntest_samples = len(test_generator.filenames)
predict_size_test = int(math.ceil(ntest_samples / batch_size))
```

```
Found 1447 images belonging to 9 classes.
```

Display the image

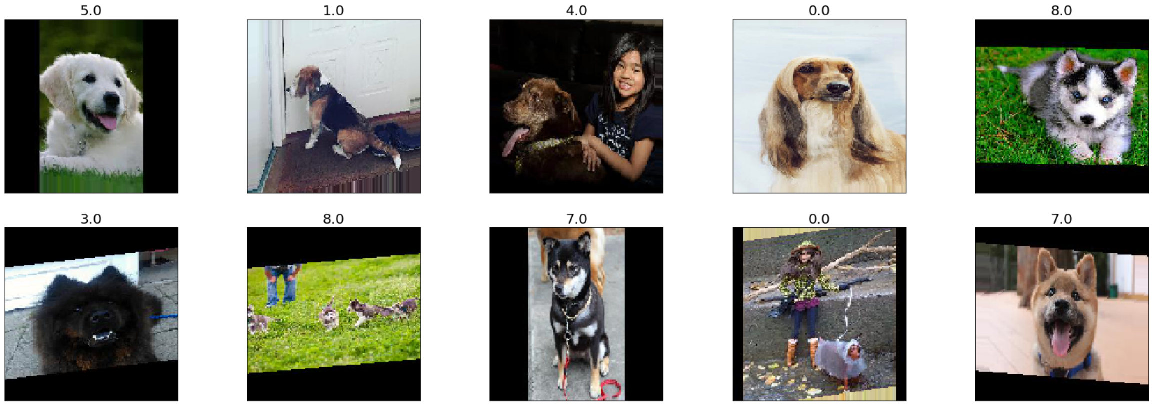
In [30]:

```
def disp_image(im):
    if (len(im.shape) == 2):
        # Gray scale image
        plt.imshow(im, cmap='gray')
    else:
        # Color image.
        im1 = (im-np.min(im))/(np.max(im)-np.min(im))*255
        im1 = im1.astype(np.uint8)
        plt.imshow(im1)
    # Remove axis ticks
    plt.xticks([])
    plt.yticks([])
```

In [44]:

```
train_generator_display=train_datagen.flow_from_directory(
                            train_data_dir,
                            target_size=(nrow,ncol),
                            batch_size=batch_size,
                            class_mode='binary',shuffle=True)
X,y=train_generator_display.next()
plt.figure(figsize=(30,10))
nplot = 10
for i in range(nplot):
    plt.subplot(2,5,i+1)
    plt.title(y[i],fontsize=20)
    disp_image(X[i,:,:,:])
```

```
Found 13599 images belonging to 9 classes.
```



Get bottleneck features

This took a long time since the input scale is large.
Use predict_generator to get bottleneck features and save them for later use.

In [18]:

```
#generate bottle neck features
bottleneck_features_train = base_model.predict_generator(train_generator, predic
t_size_train)

#save bottle neck features for training data
np.save('Images/bottleneck_features_train2.npy', bottleneck_features_train)
```

In [19]:

```
bottleneck_features_test = base_model.predict_generator(test_generator, predict_
size_test)

#save bottle neck features for test data
np.save('Images/bottleneck_features_test2.npy', bottleneck_features_test)
```

Create a different data generator for top layers

Set class_mode='catrgorical' this time.

In [21]:

```
from keras.utils.np_utils import to_categorical
datagen_top = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
batch_size = 32
train_generator_top = datagen_top.flow_from_directory(
         train_data_dir,
         target_size=(nrow,ncol),
         batch_size=batch_size,
         class_mode='categorical',
         shuffle=False)

nb_train_samples = len(train_generator_top.filenames)
num_classes = len(train_generator_top.class_indices)

train_data = np.load('Images/bottleneck_features_train2.npy')

#get the class lebels for the training data, in the original order
train_labels = train_generator_top.classes

#convert the training labels to categorical vectors
train_labels = to_categorical(train_labels, num_classes=num_classes)
```

```
Found 13599 images belonging to 9 classes.
```

In [22]:

```
test_generator_top = datagen_top.flow_from_directory(
         test_data_dir,
         target_size=(nrow,ncol),
         batch_size=batch_size,
         class_mode='categorical',
         shuffle=False)

nb_test_samples = len(test_generator_top.filenames)

test_data = np.load('Images/bottleneck_features_test2.npy')

test_labels = test_generator_top.classes
test_labels = to_categorical(test_labels, num_classes=num_classes)
```

```
Found 1447 images belonging to 9 classes.
```

```
In [23]:
```

```
train_data.shape[1:]
```

```
Out[23]:

(4, 4, 512)
```

Build top layers

We use relu and l2 regularizer for hidden layers, and softmax for output layer.

```
In [24]:
```

```
from keras import regularizers
modelVGG16=Sequential()
modelVGG16.add(Flatten(input_shape=train_data.shape[1:]))
modelVGG16.add(Dense(256, activation='relu'))
modelVGG16.add(Dropout(0.5))
modelVGG16.add(Dense(64,kernel_regularizer=regularizers.l2(0.05)))
modelVGG16.add(Dropout(0.5))
modelVGG16.add(Dense(num_classes, activation='softmax'))
modelVGG16.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
flatten_1 (Flatten)          (None, 8192)              0         
_________________________________________________________________
dense_1 (Dense)              (None, 256)               2097408   
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0         
_________________________________________________________________
dense_2 (Dense)              (None, 64)                16448     
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0         
_________________________________________________________________
dense_3 (Dense)              (None, 9)                 585       
=================================================================
Total params: 2,114,441
Trainable params: 2,114,441
Non-trainable params: 0
_________________________________________________________________
```

Compile modelVGG16 and train the model.
We use ModelCheckerpoint to save best model during training, and TerminateOnNan to stop training if loss is Nan.

In [25]:

```
from keras.callbacks import TerminateOnNaN, EarlyStopping, ModelCheckpoint
opt=optimizers.Adam(lr=1e-5, beta_1=0.9, beta_2=0.999, epsilon=1e-08, decay=0.0)
modelVGG16.compile(optimizer=opt,
              loss='categorical_crossentropy', metrics=['accuracy'])
epochs=100
top_model_weights_path = 'bottleneck_model_1.h5'
checkpointer = ModelCheckpoint(filepath=top_model_weights_path,
                               verbose=1, save_best_only=True)
EarlyStopping=EarlyStopping(monitor='val_loss', min_delta=0, patience=3, verbose
=1, mode='auto')
TerminateOnNaN=TerminateOnNaN()

history = modelVGG16.fit(train_data, train_labels,
          epochs=epochs,
          batch_size=batch_size,callbacks=[checkpointer,TerminateOnNaN],verbose=
1,
          validation_data=(test_data, test_labels))
```

```
Train on 13599 samples, validate on 1447 samples
Epoch 1/100
13536/13599 [============================>.] - ETA: 0s - loss: 7.404
3 - acc: 0.1512Epoch 00000: val_loss improved from inf to 6.98716, s
aving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 7.4032 -
acc: 0.1512 - val_loss: 6.9872 - val_acc: 0.1852
Epoch 2/100
13568/13599 [============================>.] - ETA: 0s - loss: 6.820
9 - acc: 0.2064Epoch 00001: val_loss improved from 6.98716 to 6.6315
0, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 6.8206 -
acc: 0.2066 - val_loss: 6.6315 - val_acc: 0.2377
Epoch 3/100
13536/13599 [============================>.] - ETA: 0s - loss: 6.456
5 - acc: 0.2467Epoch 00002: val_loss improved from 6.63150 to 6.2872
0, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 20s - loss: 6.4560 -
acc: 0.2468 - val_loss: 6.2872 - val_acc: 0.2764
Epoch 4/100
13536/13599 [============================>.] - ETA: 0s - loss: 6.097
1 - acc: 0.2829Epoch 00003: val_loss improved from 6.28720 to 5.9388
8, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 6.0967 -
acc: 0.2826 - val_loss: 5.9389 - val_acc: 0.3041
Epoch 5/100
13536/13599 [============================>.] - ETA: 0s - loss: 5.776
5 - acc: 0.3129Epoch 00004: val_loss improved from 5.93888 to 5.6231
6, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 20s - loss: 5.7756 -
acc: 0.3134 - val_loss: 5.6232 - val_acc: 0.3207
Epoch 6/100
13536/13599 [============================>.] - ETA: 0s - loss: 5.465
4 - acc: 0.3344Epoch 00005: val_loss improved from 5.62316 to 5.3259
8, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 21s - loss: 5.4656 -
acc: 0.3339 - val_loss: 5.3260 - val_acc: 0.3525
Epoch 7/100
13536/13599 [============================>.] - ETA: 0s - loss: 5.173
8 - acc: 0.3615Epoch 00006: val_loss improved from 5.32598 to 5.0582
3, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 20s - loss: 5.1734 -
acc: 0.3616 - val_loss: 5.0582 - val_acc: 0.3753
Epoch 8/100
13568/13599 [============================>.] - ETA: 0s - loss: 4.915
1 - acc: 0.3765Epoch 00007: val_loss improved from 5.05823 to 4.8088
1, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 20s - loss: 4.9151 -
acc: 0.3765 - val_loss: 4.8088 - val_acc: 0.3836
Epoch 9/100
13536/13599 [============================>.] - ETA: 0s - loss: 4.674
0 - acc: 0.3913Epoch 00008: val_loss improved from 4.80881 to 4.5699
4, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 18s - loss: 4.6728 -
acc: 0.3919 - val_loss: 4.5699 - val_acc: 0.4098
Epoch 10/100
13568/13599 [============================>.] - ETA: 0s - loss: 4.439
2 - acc: 0.4083Epoch 00009: val_loss improved from 4.56994 to 4.3685
1, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 21s - loss: 4.4387 -
acc: 0.4085 - val_loss: 4.3685 - val_acc: 0.4147
```

```
Epoch 11/100
13568/13599 [============================>.] - ETA: 0s - loss: 4.248
5 - acc: 0.4179Epoch 00010: val_loss improved from 4.36851 to 4.1849
0, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 4.2483 -
acc: 0.4178 - val_loss: 4.1849 - val_acc: 0.4223
Epoch 12/100
13536/13599 [============================>.] - ETA: 0s - loss: 4.051
0 - acc: 0.4352Epoch 00011: val_loss improved from 4.18490 to 3.9992
6, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 4.0503 -
acc: 0.4353 - val_loss: 3.9993 - val_acc: 0.4223
Epoch 13/100
13568/13599 [============================>.] - ETA: 0s - loss: 3.876
6 - acc: 0.4458Epoch 00012: val_loss improved from 3.99926 to 3.8207
8, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 3.8763 -
acc: 0.4458 - val_loss: 3.8208 - val_acc: 0.4471
Epoch 14/100
13568/13599 [============================>.] - ETA: 0s - loss: 3.706
8 - acc: 0.4618Epoch 00013: val_loss improved from 3.82078 to 3.6768
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 3.7062 -
acc: 0.4620 - val_loss: 3.6768 - val_acc: 0.4534
Epoch 15/100
13568/13599 [============================>.] - ETA: 0s - loss: 3.563
1 - acc: 0.4631Epoch 00014: val_loss improved from 3.67682 to 3.5176
5, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 3.5632 -
acc: 0.4632 - val_loss: 3.5177 - val_acc: 0.4603
Epoch 16/100
13568/13599 [============================>.] - ETA: 0s - loss: 3.407
6 - acc: 0.4777Epoch 00015: val_loss improved from 3.51765 to 3.3980
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 3.4074 -
acc: 0.4778 - val_loss: 3.3980 - val_acc: 0.4679
Epoch 17/100
13568/13599 [============================>.] - ETA: 0s - loss: 3.284
7 - acc: 0.4878Epoch 00016: val_loss improved from 3.39802 to 3.2917
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 3.2841 -
acc: 0.4881 - val_loss: 3.2917 - val_acc: 0.4616
Epoch 18/100
13568/13599 [============================>.] - ETA: 0s - loss: 3.168
6 - acc: 0.4946Epoch 00017: val_loss improved from 3.29172 to 3.1815
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 3.1680 -
acc: 0.4948 - val_loss: 3.1815 - val_acc: 0.4679
Epoch 19/100
13536/13599 [============================>.] - ETA: 0s - loss: 3.045
3 - acc: 0.5062Epoch 00018: val_loss improved from 3.18152 to 3.0664
8, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 3.0453 -
acc: 0.5059 - val_loss: 3.0665 - val_acc: 0.4810
Epoch 20/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.953
9 - acc: 0.5079Epoch 00019: val_loss improved from 3.06648 to 2.9850
8, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 2.9542 -
acc: 0.5077 - val_loss: 2.9851 - val_acc: 0.4789
Epoch 21/100
```

```
13536/13599 [============================>.] - ETA: 0s - loss: 2.856
9 - acc: 0.5168Epoch 00020: val_loss improved from 2.98508 to 2.9006
7, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.8566 -
acc: 0.5169 - val_loss: 2.9007 - val_acc: 0.4872
Epoch 22/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.762
7 - acc: 0.5255Epoch 00021: val_loss improved from 2.90067 to 2.8169
3, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.7628 -
acc: 0.5256 - val_loss: 2.8169 - val_acc: 0.4879
Epoch 23/100
13568/13599 [============================>.] - ETA: 0s - loss: 2.684
9 - acc: 0.5279Epoch 00022: val_loss improved from 2.81693 to 2.7394
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.6848 -
acc: 0.5280 - val_loss: 2.7394 - val_acc: 0.4934
Epoch 24/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.595
7 - acc: 0.5368Epoch 00023: val_loss improved from 2.73942 to 2.6760
1, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.5957 -
acc: 0.5369 - val_loss: 2.6760 - val_acc: 0.4948
Epoch 25/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.521
3 - acc: 0.5446Epoch 00024: val_loss improved from 2.67601 to 2.6123
5, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.5205 -
acc: 0.5450 - val_loss: 2.6123 - val_acc: 0.4865
Epoch 26/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.454
9 - acc: 0.5486Epoch 00025: val_loss improved from 2.61235 to 2.5410
3, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.4546 -
acc: 0.5486 - val_loss: 2.5410 - val_acc: 0.5031
Epoch 27/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.391
4 - acc: 0.5519Epoch 00026: val_loss improved from 2.54103 to 2.5001
5, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.3901 -
acc: 0.5526 - val_loss: 2.5001 - val_acc: 0.4976
Epoch 28/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.331
9 - acc: 0.5627Epoch 00027: val_loss improved from 2.50015 to 2.4486
4, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.3314 -
acc: 0.5629 - val_loss: 2.4486 - val_acc: 0.4997
Epoch 29/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.268
4 - acc: 0.5680Epoch 00028: val_loss improved from 2.44864 to 2.4017
3, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.2688 -
acc: 0.5684 - val_loss: 2.4017 - val_acc: 0.4955
Epoch 30/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.212
8 - acc: 0.5752Epoch 00029: val_loss improved from 2.40173 to 2.3606
6, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.2138 -
acc: 0.5746 - val_loss: 2.3607 - val_acc: 0.4983
Epoch 31/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.164
```

```
7 - acc: 0.5764Epoch 00030: val_loss improved from 2.36066 to 2.3029
1, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.1647 -
acc: 0.5765 - val_loss: 2.3029 - val_acc: 0.4997
Epoch 32/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.115
5 - acc: 0.5789Epoch 00031: val_loss improved from 2.30291 to 2.2677
6, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.1154 -
acc: 0.5791 - val_loss: 2.2678 - val_acc: 0.5003
Epoch 33/100
13568/13599 [============================>.] - ETA: 0s - loss: 2.067
0 - acc: 0.5872Epoch 00032: val_loss improved from 2.26776 to 2.2325
8, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.0669 -
acc: 0.5872 - val_loss: 2.2326 - val_acc: 0.5017
Epoch 34/100
13536/13599 [============================>.] - ETA: 0s - loss: 2.028
7 - acc: 0.5898Epoch 00033: val_loss improved from 2.23258 to 2.1882
6, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 2.0272 -
acc: 0.5906 - val_loss: 2.1883 - val_acc: 0.5114
Epoch 35/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.984
2 - acc: 0.5949Epoch 00034: val_loss improved from 2.18826 to 2.1567
3, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 1.9840 -
acc: 0.5951 - val_loss: 2.1567 - val_acc: 0.5066
Epoch 36/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.940
3 - acc: 0.5991Epoch 00035: val_loss improved from 2.15673 to 2.1283
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 1.9405 -
acc: 0.5990 - val_loss: 2.1283 - val_acc: 0.5114
Epoch 37/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.906
2 - acc: 0.6068Epoch 00036: val_loss improved from 2.12832 to 2.0946
1, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 20s - loss: 1.9060 -
acc: 0.6068 - val_loss: 2.0946 - val_acc: 0.5107
Epoch 38/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.873
2 - acc: 0.6113Epoch 00037: val_loss improved from 2.09461 to 2.0540
0, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 20s - loss: 1.8733 -
acc: 0.6109 - val_loss: 2.0540 - val_acc: 0.5162
Epoch 39/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.843
8 - acc: 0.6128Epoch 00038: val_loss improved from 2.05400 to 2.0510
6, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 1.8441 -
acc: 0.6126 - val_loss: 2.0511 - val_acc: 0.5114
Epoch 40/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.801
4 - acc: 0.6209Epoch 00039: val_loss improved from 2.05106 to 2.0120
7, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.8015 -
acc: 0.6211 - val_loss: 2.0121 - val_acc: 0.5204
Epoch 41/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.768
8 - acc: 0.6226Epoch 00040: val_loss improved from 2.01207 to 1.9922
```

```
7, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 18s - loss: 1.7690 -
acc: 0.6223 - val_loss: 1.9923 - val_acc: 0.5169
Epoch 42/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.743
0 - acc: 0.6240Epoch 00041: val_loss improved from 1.99227 to 1.9572
6, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 18s - loss: 1.7432 -
acc: 0.6240 - val_loss: 1.9573 - val_acc: 0.5204
Epoch 43/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.713
3 - acc: 0.6271Epoch 00042: val_loss improved from 1.95726 to 1.9491
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 18s - loss: 1.7128 -
acc: 0.6274 - val_loss: 1.9491 - val_acc: 0.5176
Epoch 44/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.681
1 - acc: 0.6334Epoch 00043: val_loss improved from 1.94912 to 1.9413
5, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.6815 -
acc: 0.6332 - val_loss: 1.9414 - val_acc: 0.5128
Epoch 45/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.654
4 - acc: 0.6381Epoch 00044: val_loss improved from 1.94135 to 1.9003
0, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.6546 -
acc: 0.6380 - val_loss: 1.9003 - val_acc: 0.5218
Epoch 46/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.628
1 - acc: 0.6373Epoch 00045: val_loss improved from 1.90030 to 1.8692
3, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.6272 -
acc: 0.6375 - val_loss: 1.8692 - val_acc: 0.5280
Epoch 47/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.597
1 - acc: 0.6497Epoch 00046: val_loss improved from 1.86923 to 1.8623
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.5975 -
acc: 0.6492 - val_loss: 1.8623 - val_acc: 0.5225
Epoch 48/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.576
1 - acc: 0.6520Epoch 00047: val_loss improved from 1.86232 to 1.8546
4, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.5758 -
acc: 0.6520 - val_loss: 1.8546 - val_acc: 0.5225
Epoch 49/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.552
7 - acc: 0.6520Epoch 00048: val_loss improved from 1.85464 to 1.8379
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 1.5523 -
acc: 0.6522 - val_loss: 1.8379 - val_acc: 0.5225
Epoch 50/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.532
8 - acc: 0.6550Epoch 00049: val_loss did not improve
13599/13599 [==============================] - 20s - loss: 1.5322 -
acc: 0.6555 - val_loss: 1.8399 - val_acc: 0.5176
Epoch 51/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.514
0 - acc: 0.6596Epoch 00050: val_loss improved from 1.83792 to 1.7980
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.5134 -
```

```
acc: 0.6599 - val_loss: 1.7980 - val_acc: 0.5238
Epoch 52/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.491
4 - acc: 0.6662Epoch 00051: val_loss improved from 1.79802 to 1.7935
1, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 1.4911 -
acc: 0.6663 - val_loss: 1.7935 - val_acc: 0.5245
Epoch 53/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.470
1 - acc: 0.6644Epoch 00052: val_loss improved from 1.79351 to 1.7661
1, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 1.4697 -
acc: 0.6644 - val_loss: 1.7661 - val_acc: 0.5342
Epoch 54/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.450
8 - acc: 0.6714Epoch 00053: val_loss did not improve
13599/13599 [==============================] - 20s - loss: 1.4505 -
acc: 0.6715 - val_loss: 1.7762 - val_acc: 0.5211
Epoch 55/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.430
4 - acc: 0.6677Epoch 00054: val_loss did not improve
13599/13599 [==============================] - 19s - loss: 1.4297 -
acc: 0.6679 - val_loss: 1.7681 - val_acc: 0.5232
Epoch 56/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.409
3 - acc: 0.6774Epoch 00055: val_loss improved from 1.76611 to 1.7371
2, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.4099 -
acc: 0.6772 - val_loss: 1.7371 - val_acc: 0.5321
Epoch 57/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.389
0 - acc: 0.6818Epoch 00056: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 1.3892 -
acc: 0.6818 - val_loss: 1.7543 - val_acc: 0.5121
Epoch 58/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.381
1 - acc: 0.6788Epoch 00057: val_loss improved from 1.73712 to 1.7306
3, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 18s - loss: 1.3820 -
acc: 0.6785 - val_loss: 1.7306 - val_acc: 0.5287
Epoch 59/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.358
4 - acc: 0.6865Epoch 00058: val_loss improved from 1.73063 to 1.7180
0, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 20s - loss: 1.3579 -
acc: 0.6867 - val_loss: 1.7180 - val_acc: 0.5342
Epoch 60/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.340
6 - acc: 0.6868Epoch 00059: val_loss improved from 1.71800 to 1.7134
1, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.3403 -
acc: 0.6868 - val_loss: 1.7134 - val_acc: 0.5238
Epoch 61/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.329
2 - acc: 0.6936Epoch 00060: val_loss improved from 1.71341 to 1.7029
7, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.3295 -
acc: 0.6933 - val_loss: 1.7030 - val_acc: 0.5294
Epoch 62/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.315
0 - acc: 0.6910Epoch 00061: val_loss improved from 1.70297 to 1.6826
```

```
6, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.3150 -
acc: 0.6907 - val_loss: 1.6827 - val_acc: 0.5390
Epoch 63/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.288
8 - acc: 0.6994Epoch 00062: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 1.2887 -
acc: 0.6993 - val_loss: 1.6939 - val_acc: 0.5280
Epoch 64/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.275
0 - acc: 0.7061Epoch 00063: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 1.2741 -
acc: 0.7064 - val_loss: 1.6907 - val_acc: 0.5280
Epoch 65/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.264
1 - acc: 0.7056Epoch 00064: val_loss did not improve
13599/13599 [==============================] - 19s - loss: 1.2640 -
acc: 0.7056 - val_loss: 1.7048 - val_acc: 0.5176
Epoch 66/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.247
7 - acc: 0.7040Epoch 00065: val_loss improved from 1.68266 to 1.6481
3, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 19s - loss: 1.2477 -
acc: 0.7039 - val_loss: 1.6481 - val_acc: 0.5404
Epoch 67/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.238
7 - acc: 0.7070Epoch 00066: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 1.2382 -
acc: 0.7074 - val_loss: 1.6561 - val_acc: 0.5390
Epoch 68/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.225
9 - acc: 0.7137Epoch 00067: val_loss did not improve
13599/13599 [==============================] - 17s - loss: 1.2259 -
acc: 0.7137 - val_loss: 1.6602 - val_acc: 0.5273
Epoch 69/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.207
9 - acc: 0.7159Epoch 00068: val_loss improved from 1.64813 to 1.6188
3, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 1.2078 -
acc: 0.7158 - val_loss: 1.6188 - val_acc: 0.5411
Epoch 70/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.204
6 - acc: 0.7163Epoch 00069: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 1.2043 -
acc: 0.7164 - val_loss: 1.6317 - val_acc: 0.5370
Epoch 71/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.183
4 - acc: 0.7196Epoch 00070: val_loss did not improve
13599/13599 [==============================] - 17s - loss: 1.1833 -
acc: 0.7197 - val_loss: 1.6251 - val_acc: 0.5349
Epoch 72/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.181
0 - acc: 0.7202Epoch 00071: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 1.1808 -
acc: 0.7204 - val_loss: 1.6346 - val_acc: 0.5287
Epoch 73/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.159
6 - acc: 0.7260Epoch 00072: val_loss did not improve
13599/13599 [==============================] - 17s - loss: 1.1601 -
acc: 0.7259 - val_loss: 1.6304 - val_acc: 0.5370
Epoch 74/100
```

```
13536/13599 [============================>.] - ETA: 0s - loss: 1.151
0 - acc: 0.7286Epoch 00073: val_loss improved from 1.61883 to 1.5933
4, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 17s - loss: 1.1515 -
acc: 0.7290 - val_loss: 1.5933 - val_acc: 0.5487
Epoch 75/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.137
1 - acc: 0.7325Epoch 00074: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 1.1366 -
acc: 0.7327 - val_loss: 1.6233 - val_acc: 0.5308
Epoch 76/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.132
0 - acc: 0.7286Epoch 00075: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 1.1321 -
acc: 0.7286 - val_loss: 1.6120 - val_acc: 0.5349
Epoch 77/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.123
2 - acc: 0.7349Epoch 00076: val_loss did not improve
13599/13599 [==============================] - 17s - loss: 1.1226 -
acc: 0.7351 - val_loss: 1.6006 - val_acc: 0.5397
Epoch 78/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.099
0 - acc: 0.7422Epoch 00077: val_loss did not improve
13599/13599 [==============================] - 19s - loss: 1.0985 -
acc: 0.7423 - val_loss: 1.6066 - val_acc: 0.5370
Epoch 79/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.097
1 - acc: 0.7436Epoch 00078: val_loss did not improve
13599/13599 [==============================] - 19s - loss: 1.0971 -
acc: 0.7436 - val_loss: 1.6061 - val_acc: 0.5287
Epoch 80/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.084
0 - acc: 0.7459Epoch 00079: val_loss did not improve
13599/13599 [==============================] - 19s - loss: 1.0844 -
acc: 0.7456 - val_loss: 1.6055 - val_acc: 0.5356
Epoch 81/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.076
4 - acc: 0.7429Epoch 00080: val_loss did not improve
13599/13599 [==============================] - 19s - loss: 1.0761 -
acc: 0.7428 - val_loss: 1.6015 - val_acc: 0.5356
Epoch 82/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.065
8 - acc: 0.7483Epoch 00081: val_loss improved from 1.59334 to 1.5921
6, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 18s - loss: 1.0663 -
acc: 0.7481 - val_loss: 1.5922 - val_acc: 0.5370
Epoch 83/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.058
6 - acc: 0.7514Epoch 00082: val_loss did not improve
13599/13599 [==============================] - 19s - loss: 1.0585 -
acc: 0.7514 - val_loss: 1.6075 - val_acc: 0.5314
Epoch 84/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.050
1 - acc: 0.7546Epoch 00083: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 1.0503 -
acc: 0.7545 - val_loss: 1.5966 - val_acc: 0.5342
Epoch 85/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.040
1 - acc: 0.7587Epoch 00084: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 1.0400 -
acc: 0.7586 - val_loss: 1.5923 - val_acc: 0.5411
```

```
Epoch 86/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.035
4 - acc: 0.7552Epoch 00085: val_loss improved from 1.59216 to 1.5785
3, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 20s - loss: 1.0352 -
acc: 0.7554 - val_loss: 1.5785 - val_acc: 0.5404
Epoch 87/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.021
3 - acc: 0.7554Epoch 00086: val_loss did not improve
13599/13599 [==============================] - 16s - loss: 1.0215 -
acc: 0.7555 - val_loss: 1.5973 - val_acc: 0.5321
Epoch 88/100
13568/13599 [============================>.] - ETA: 0s - loss: 1.006
3 - acc: 0.7622Epoch 00087: val_loss improved from 1.57853 to 1.5761
5, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 16s - loss: 1.0063 -
acc: 0.7623 - val_loss: 1.5761 - val_acc: 0.5384
Epoch 89/100
13536/13599 [============================>.] - ETA: 0s - loss: 1.007
6 - acc: 0.7652Epoch 00088: val_loss did not improve
13599/13599 [==============================] - 16s - loss: 1.0074 -
acc: 0.7652 - val_loss: 1.5979 - val_acc: 0.5328
Epoch 90/100
13536/13599 [============================>.] - ETA: 0s - loss: 0.995
9 - acc: 0.7677Epoch 00089: val_loss did not improve
13599/13599 [==============================] - 16s - loss: 0.9961 -
acc: 0.7674 - val_loss: 1.5982 - val_acc: 0.5328
Epoch 91/100
13536/13599 [============================>.] - ETA: 0s - loss: 0.985
3 - acc: 0.7687Epoch 00090: val_loss did not improve
13599/13599 [==============================] - 16s - loss: 0.9856 -
acc: 0.7687 - val_loss: 1.5935 - val_acc: 0.5314
Epoch 92/100
13568/13599 [============================>.] - ETA: 0s - loss: 0.982
9 - acc: 0.7684Epoch 00091: val_loss did not improve
13599/13599 [==============================] - 17s - loss: 0.9834 -
acc: 0.7681 - val_loss: 1.6045 - val_acc: 0.5314
Epoch 93/100
13536/13599 [============================>.] - ETA: 0s - loss: 0.976
3 - acc: 0.7752Epoch 00092: val_loss improved from 1.57615 to 1.5634
6, saving model to bottleneck_model_1.h5
13599/13599 [==============================] - 21s - loss: 0.9765 -
acc: 0.7751 - val_loss: 1.5635 - val_acc: 0.5418
Epoch 94/100
13536/13599 [============================>.] - ETA: 0s - loss: 0.963
7 - acc: 0.7736Epoch 00093: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 0.9636 -
acc: 0.7738 - val_loss: 1.5925 - val_acc: 0.5342
Epoch 95/100
13568/13599 [============================>.] - ETA: 0s - loss: 0.956
3 - acc: 0.7804Epoch 00094: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 0.9563 -
acc: 0.7806 - val_loss: 1.5762 - val_acc: 0.5363
Epoch 96/100
13536/13599 [============================>.] - ETA: 0s - loss: 0.952
9 - acc: 0.7798Epoch 00095: val_loss did not improve
13599/13599 [==============================] - 19s - loss: 0.9526 -
acc: 0.7798 - val_loss: 1.5837 - val_acc: 0.5335
Epoch 97/100
13536/13599 [============================>.] - ETA: 0s - loss: 0.947
9 - acc: 0.7815Epoch 00096: val_loss did not improve
```

```
13599/13599 [==============================] - 19s - loss: 0.9479 -
acc: 0.7815 - val_loss: 1.5688 - val_acc: 0.5384
Epoch 98/100
13568/13599 [============================>.] - ETA: 0s - loss: 0.936
4 - acc: 0.7846Epoch 00097: val_loss did not improve
13599/13599 [==============================] - 19s - loss: 0.9363 -
acc: 0.7847 - val_loss: 1.5806 - val_acc: 0.5335
Epoch 99/100
13536/13599 [============================>.] - ETA: 0s - loss: 0.927
3 - acc: 0.7926Epoch 00098: val_loss did not improve
13599/13599 [==============================] - 18s - loss: 0.9271 -
acc: 0.7927 - val_loss: 1.5834 - val_acc: 0.5349
Epoch 100/100
13568/13599 [============================>.] - ETA: 0s - loss: 0.923
1 - acc: 0.7886Epoch 00099: val_loss did not improve
13599/13599 [==============================] - 20s - loss: 0.9232 -
acc: 0.7884 - val_loss: 1.6011 - val_acc: 0.5328
```

Save the final weights in case it's needed.

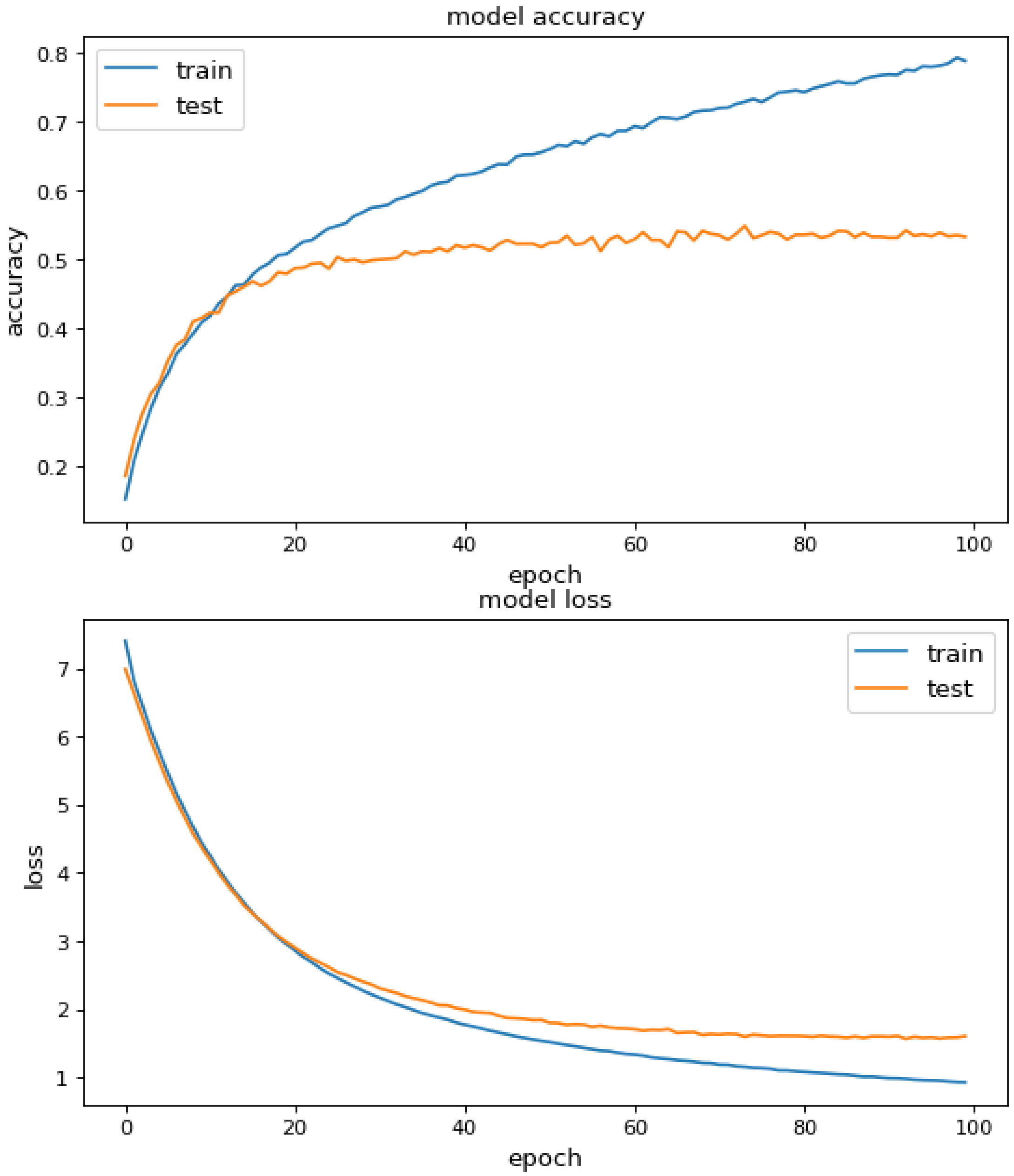
In [26]:

```
modelVGG16.save('bottleneck_model_1_final.h5')
```

In [27]:

```
plt.figure(figsize=(8,9.5),dpi=80, facecolor='w', edgecolor='k')
# summarize history for accuracy
plt.subplot(211)
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy',fontsize=12)
plt.ylabel('accuracy',fontsize=12)
plt.xlabel('epoch',fontsize=12)
plt.legend(['train', 'test'], loc='best',fontsize=12)

# summarize history for loss
plt.subplot(212)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss',fontsize=12)
plt.ylabel('loss',fontsize=12)
plt.xlabel('epoch',fontsize=12)
plt.legend(['train', 'test'], loc='best',fontsize=12)
plt.show()
```



Loss and accuracy of test data saturated as training goes on.

We've tried other models with different regularizer penalties or higher dropout, but the results weren't as good as this model.

References

1. Building Powerful Image Classification Models Using Very Little Data (https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html)

2. Using Bottleneck Features for Multi-Class Classification in Keras and TensorFlow (http://www.codesofinterest.com/2017/08/bottleneck-features-multi-class-classification-keras.html)

3. Dog Breed Prediction with Convolutional Neural Networks (https://github.com/jeremyjordan/dog-breed-classifier/blob/master/dog_app.ipynb)

In [ ]:

```

```