CS4487 - Machine Learning

Lecture 10a - Deep Learning 2 ¶

Dr. Antoni B. Chan

Dept. of Computer Science, City University of Hong Kong

Outline

- Image Classification and Deep Architectures
- Unsupervised Learning

In [1]:
```
# setup
%matplotlib inline
import IPython.core.display         # setup output image format (Chrome works best)
IPython.core.display.set_matplotlib_formats("svg")
import matplotlib.pyplot as plt
import matplotlib
from numpy import *
from sklearn import *
from scipy import stats

rbow = plt.get_cmap('rainbow')
```

In [2]:
```
# use TensorFlow backend
%env KERAS_BACKEND=tensorflow
from keras.models import Sequential, Model
from keras.layers import Dense, Activation, Dropout, Conv2D, F
latten, \
                Input, MaxPooling2D, UpSampling2D, Lambda, Res
hape, BatchNormalization, \
                GlobalAveragePooling2D
import keras
import tensorflow
import logging
logging.basicConfig()
import struct

# use channels first representation for images
from keras import backend as K
K.set_image_data_format('channels_first')

from keras.callbacks import TensorBoard
```

```
env: KERAS_BACKEND=tensorflow

Using TensorFlow backend.
```

In [3]:
```
def plot_history(history):
    fig, ax1 = plt.subplots()

    ax1.plot(history.history['loss'], 'r', label="training los
s ({:.6f})".format(history.history['loss'][-1]))
    ax1.plot(history.history['val_loss'], 'r--', label="valida
tion loss ({:.6f})".format(history.history['val_loss'][-1]))
    ax1.grid(True)
    ax1.set_xlabel('iteration')
    ax1.legend(loc="best", fontsize=9)
    ax1.set_ylabel('loss', color='r')
    ax1.tick_params('y', colors='r')

    if 'acc' in history.history:
        ax2 = ax1.twinx()

        ax2.plot(history.history['acc'], 'b', label="training
acc ({:.4f})".format(history.history['acc'][-1]))
        ax2.plot(history.history['val_acc'], 'b--', label="val
idation acc ({:.4f})".format(history.history['val_acc'][-1]))

        ax2.legend(loc="best", fontsize=9)
        ax2.set_ylabel('acc', color='b')
        ax2.tick_params('y', colors='b')
```

In [4]:

```
def show_imgs(W_list, nc=10, highlight_green=None, highlight_r
ed=None, titles=None):
    nfilter = len(W_list)
    nr = (nfilter - 1) // nc + 1
    for i in range(nr):
        for j in range(nc):
            idx = i * nc + j
            if idx == nfilter:
                break
            plt.subplot(nr, nc, idx + 1)
            cur_W = W_list[idx]
            if ndim(cur_W) == 2:
                plt.imshow(cur_W,cmap='gray', interpolation='n
earest')
            else:
                plt.imshow(cur_W,interpolation='nearest')

            if titles is not None:
                if isinstance(titles, str):
                    plt.title(titles % idx)
                elif isinstance(titles, list):
                    plt.title(titles[idx])
                else:
                    plt.title(titles)

            if ((highlight_green is not None) and highlight_gr
een[idx]) or \
               ((highlight_red is not None) and highlight_red[
idx]):
                ax = plt.gca()
                if highlight_green[idx]:
                    mycol = '#00FF00'
                else:
                    mycol = 'r'
                for S in ['bottom', 'top', 'right', 'left']:
                    ax.spines[S].set_color(mycol)
                    ax.spines[S].set_lw(2.0)
                ax.xaxis.set_ticks_position('none')
                ax.yaxis.set_ticks_position('none')
                ax.set_xticks([])
                ax.set_yticks([])
            else:
                plt.gca().set_axis_off()
```

Image Classification

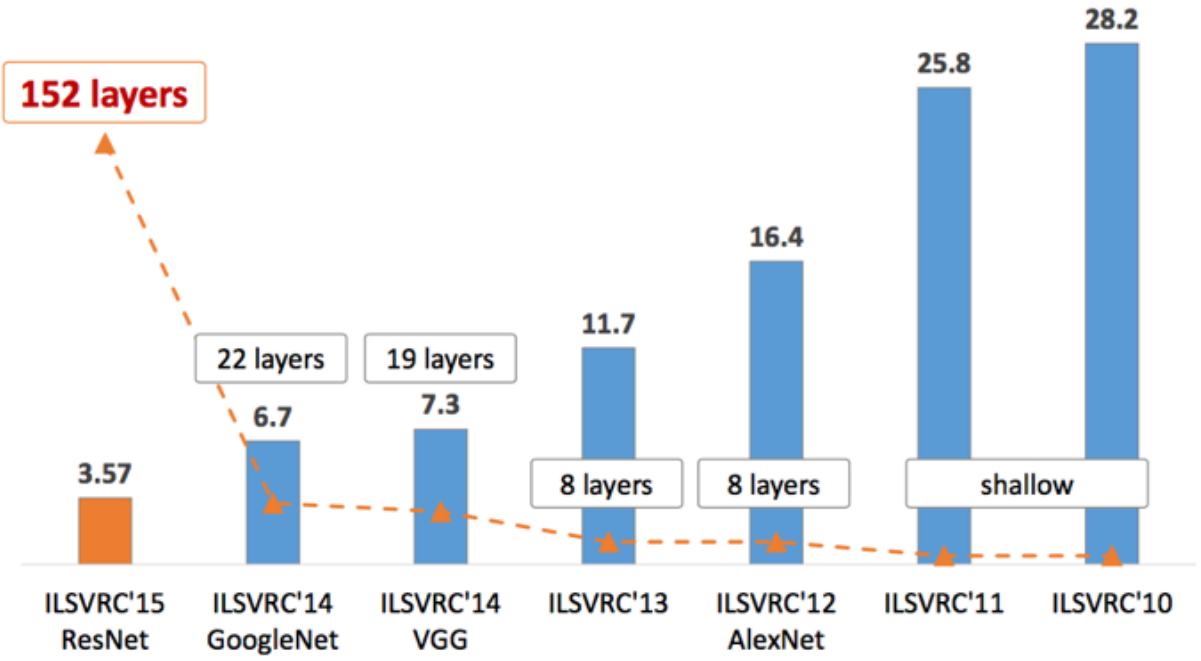
- **Goal:** given an image, predict the object class
  - typically only one main object is present
  - If enough classes are considered, then it's a generic high-level vision task

ImageNet

- ImageNet Large Scale Visual Recognition Challenge (ILSVRC)
  - 1000 image classes
  - 1.2 million images
  - Human performance: ~5.1%

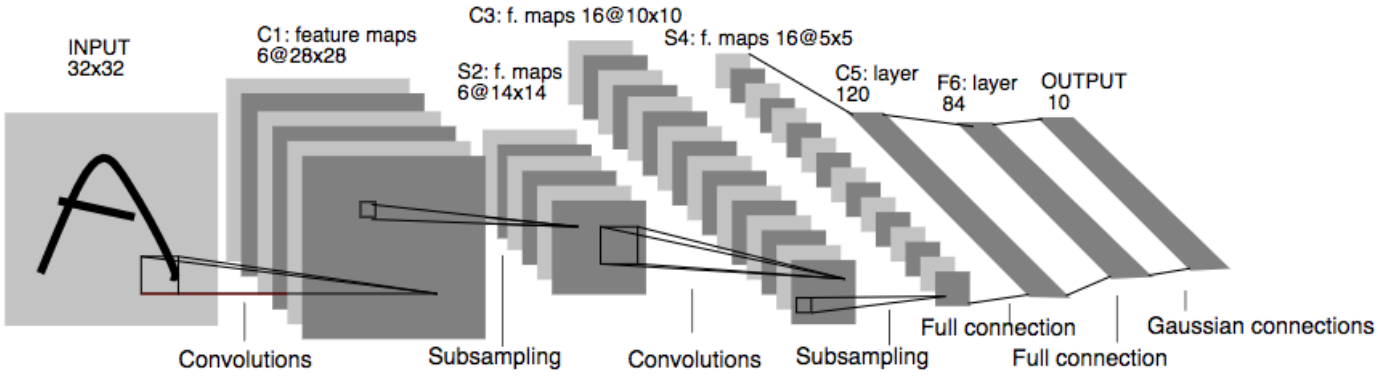
Performance of Deep Learning

- The introduction of ILSVRC coincided with the emergence of Deep Learning.
- Top-5 error rate decreased as deeper NN were developed
  - Not just deeper, but also the architecture design was smarter



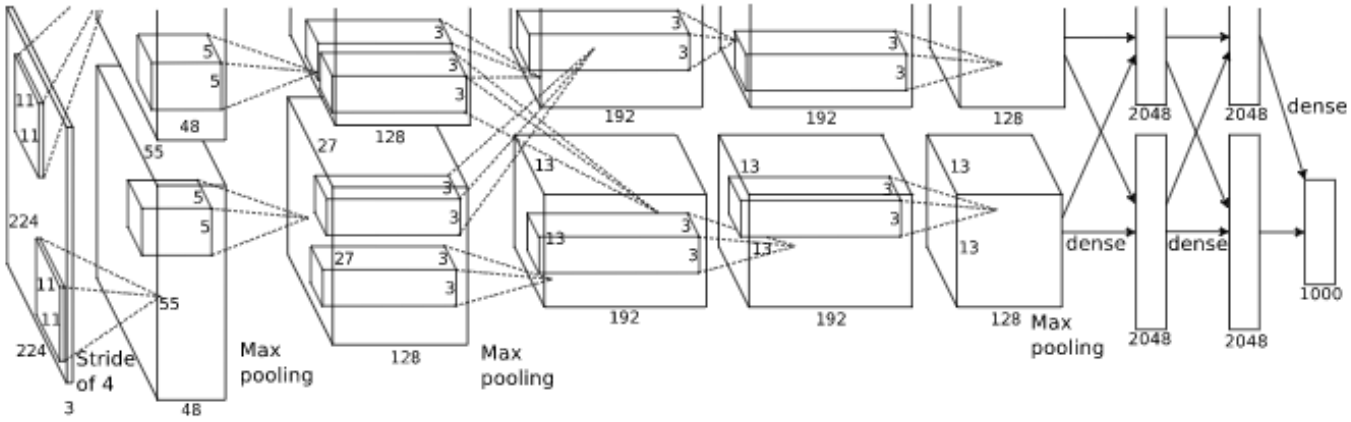
LeNet-5 (1998)

- The standard CNN architecture
  - 7 layers
  - convolutions & pooling, final fully-connected layer
- Designed for hand-written digit recongition (MNIST)



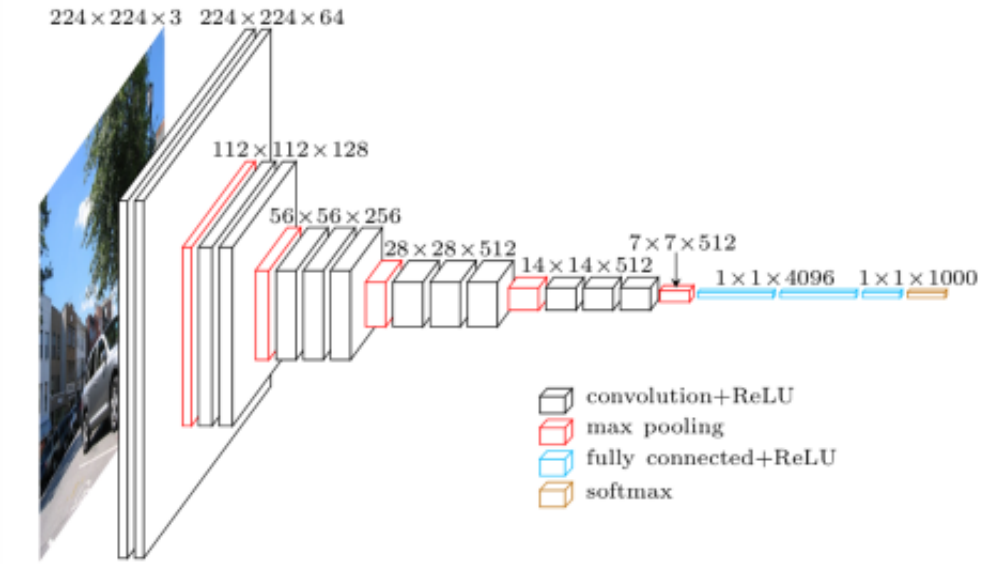
AlexNet

- Similar architecture to LeNet, but deeper (14 layers)
  - 11x11, 5x5, 3x3 convolutions
- Tricks used: DropOut, ReLU, max pooling, data augmentation, SGD momentum
- One of the first networks trained on GPU
  - (it is split in two pipelines because it was trained on 2 GPUs simultaneously)



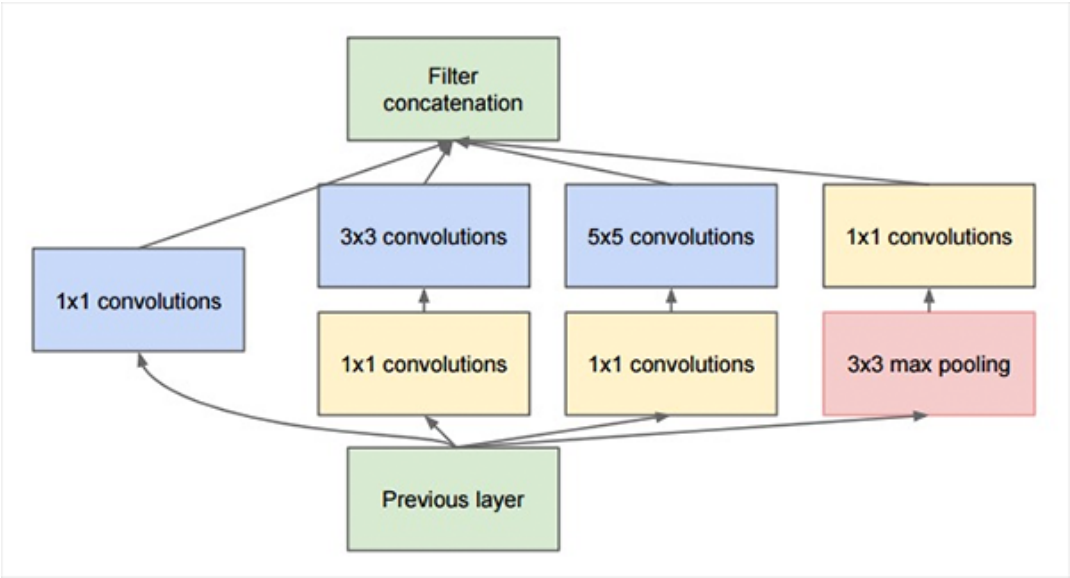
VGGnet

- Same style as LeNet and AlexNet
- Design choices:
  - only use 3x3 convolution filters (less parameters)
  - stack several convolution layers one after another, followed by pooling
    - equivalent to using a larger filter, but with less parameters.
  - number of feature channels doubles after each stage.
    - more higher-level features



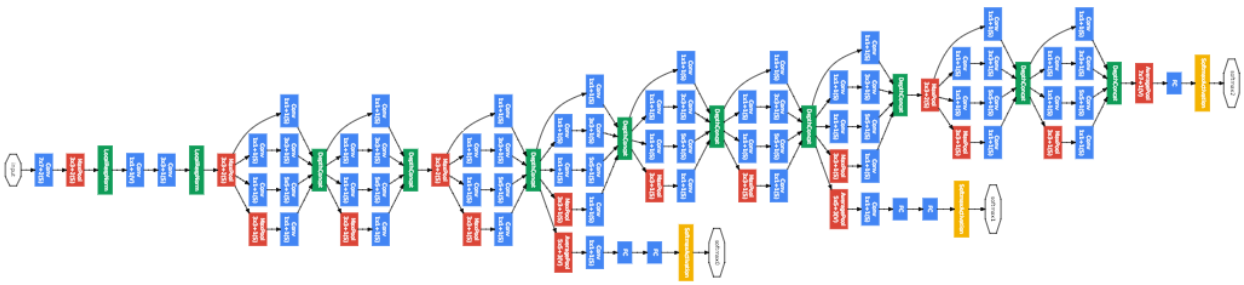
InceptionNet

- "Network-in-Network" architecture
  - Core building blocks (modules), consisting of several layers, are combined repeatedly.
- Inception Module
  - several convolution filters in parallel
    - extract features at different scales (1x1, 3x3, 5x5)
    - pool features (3x3 max)
  - features are concatenated and passed to next block.



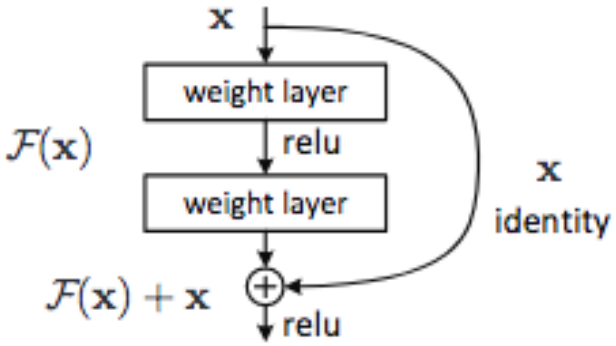
InceptionNet (V1)

- 9 inception modules, 22 layers
  - 50 convolution blocks
- Auxiliary classification tasks
  - using features in the middle of the network to perform classification
  - strengthen supervisory signals to the middle and earlier layers.



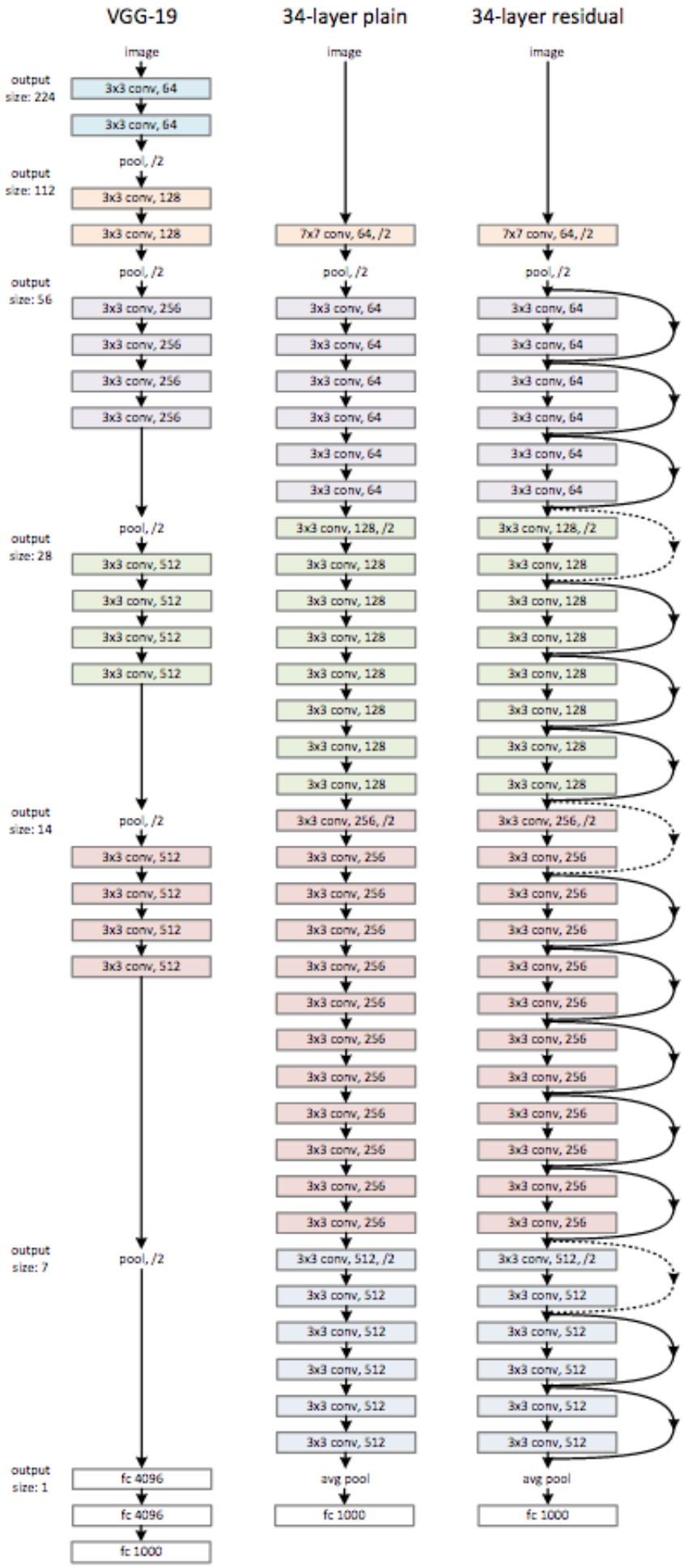
Residual Learning

- The network is learning a function (image to class)
  - build the function block-by-block
  - each block learns a residual, which is added to the previous block
    - keep all the previous information and make small changes with the residual

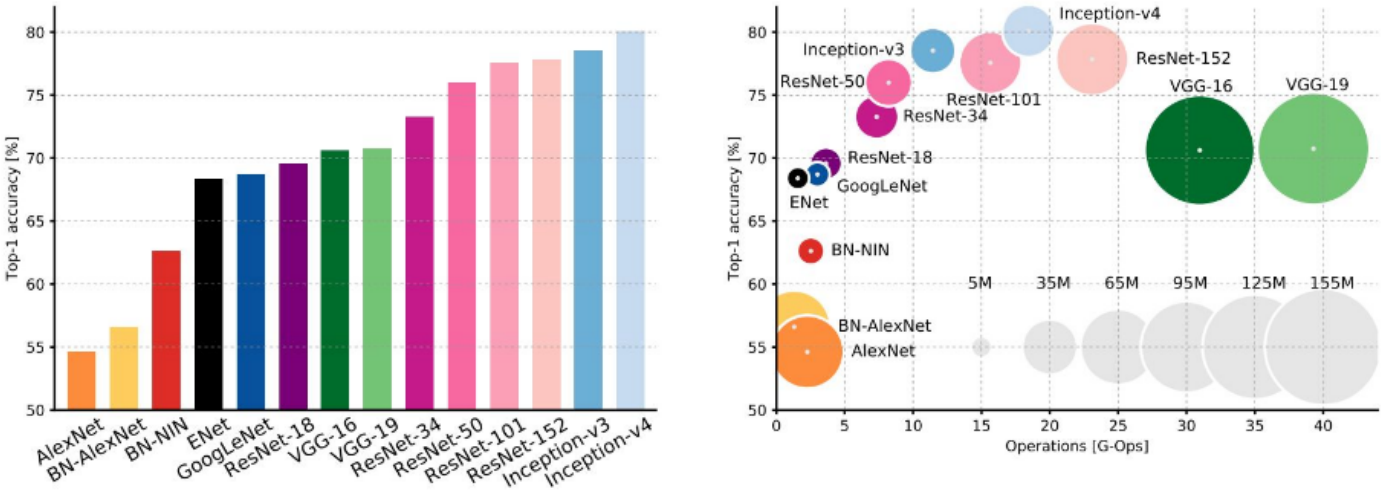


Residual Network (ResNet)

- 34 layers, 50 layers, 100 layers, 1000 layers
  - 3x3 filters
  - residual connection every two layers.


VGG-19
34-layer plain
34-layer residual
image
output size: 224
3x3 conv, 64
3x3 conv, 64
pool, /2
output size: 112
3x3 conv, 128
3x3 conv, 128
pool, /2
output size: 56
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
pool, /2
output size: 28
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
pool, /2
output size: 14
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
pool, /2
output size: 7
fc 4096
fc 4096
fc 1000
output size: 1
7x7 conv, 64, /2
pool, /2
3x3 conv, 64
3x3 conv, 128, /2
3x3 conv, 128
3x3 conv, 256, /2
3x3 conv, 256
3x3 conv, 512, /2
3x3 conv, 512
avg pool
fc 1000

Error vs. Computation vs. Number of Parameters



An Analysis of Deep Neural Network Models for Practical Applications, 2017.

Image Classifiers in Keras

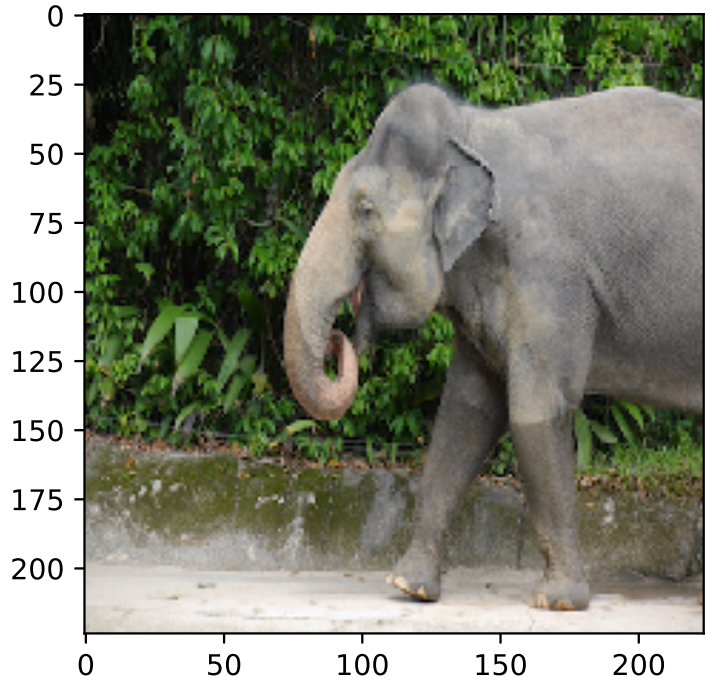
- Keras includes several pre-trained image classifier networks
  - VGG16, VGG19, ResNet50, InceptionV3, ...
  - already trained on ImageNet
- Can use these networks to:
  - perform image classification (for 1000 classes only)
  - extract image features for our own task
  - transfer learning - modify/adapt a pre-trained network for our own task

- Import the ResNet model class and support functions
- Load and pre-process the image for the network

In [59]:
```
# contains the model and support functions for ResNet50
import keras.applications.resnet50 as resnet
from keras.preprocessing import image

# load an image
img_path = 'imgs/elephant1.jpg'
img = image.load_img(img_path, target_size=(224, 224))  # ResN
et expects this size
plt.imshow(img)

# process the image for resnet
xi = image.img_to_array(img)
xi = expand_dims(xi, axis=0)
xi = resnet.preprocess_input(xi)
```



- Create an instance of the ResNet50 model
  - trained on ImageNet
  - will download the model if loading for the first time.

In [6]:
```
# create an instance of the model
model = resnet.ResNet50(weights='imagenet')
model.summary()
```

```
__________________________________________________________________
_____________________________________
Layer (type)                 Output Shape              Param #
Connected to
==================================================================
====================================
input_1 (InputLayer)         (None, 3, 224, 224)   0
__________________________________________________________________
_____________________________________
conv1_pad (ZeroPadding2D)    (None, 3, 230, 230)   0
```

```
input_1[0][0]
____________________________________________________________________________
____________________________________________
conv1 (Conv2D)                   (None, 64, 112, 112) 9472
conv1_pad[0][0]
____________________________________________________________________________
____________________________________________
bn_conv1 (BatchNormalization)    (None, 64, 112, 112) 256
conv1[0][0]
____________________________________________________________________________
____________________________________________
activation_1 (Activation)        (None, 64, 112, 112) 0
bn_conv1[0][0]
____________________________________________________________________________
____________________________________________
pool1_pad (ZeroPadding2D)        (None, 64, 114, 114) 0
activation_1[0][0]
____________________________________________________________________________
____________________________________________
max_pooling2d_1 (MaxPooling2D)   (None, 64, 56, 56)   0
pool1_pad[0][0]
____________________________________________________________________________
____________________________________________
res2a_branch2a (Conv2D)          (None, 64, 56, 56)   4160
max_pooling2d_1[0][0]
____________________________________________________________________________
____________________________________________
bn2a_branch2a (BatchNormalizati (None, 64, 56, 56)   256
res2a_branch2a[0][0]
____________________________________________________________________________
____________________________________________
activation_2 (Activation)        (None, 64, 56, 56)   0
bn2a_branch2a[0][0]
____________________________________________________________________________
____________________________________________
res2a_branch2b (Conv2D)          (None, 64, 56, 56)   36928
activation_2[0][0]
____________________________________________________________________________
____________________________________________
bn2a_branch2b (BatchNormalizati (None, 64, 56, 56)   256
res2a_branch2b[0][0]
____________________________________________________________________________
____________________________________________
activation_3 (Activation)        (None, 64, 56, 56)   0
bn2a_branch2b[0][0]
____________________________________________________________________________
____________________________________________
res2a_branch2c (Conv2D)          (None, 256, 56, 56)  16640
activation_3[0][0]
____________________________________________________________________________
____________________________________________
res2a_branch1 (Conv2D)           (None, 256, 56, 56)  16640
max_pooling2d_1[0][0]
____________________________________________________________________________
____________________________________________
bn2a_branch2c (BatchNormalizati (None, 256, 56, 56)  1024
res2a_branch2c[0][0]
```

```
__________________________________________________________________________________________
__________________________________________
bn2a_branch1 (BatchNormalizatio (None, 256, 56, 56)  1024
res2a_branch1[0][0]
__________________________________________________________________________________________
__________________________________________
add_1 (Add)                     (None, 256, 56, 56)  0
bn2a_branch2c[0][0]

bn2a_branch1[0][0]
__________________________________________________________________________________________
__________________________________________
activation_4 (Activation)       (None, 256, 56, 56)  0
add_1[0][0]
__________________________________________________________________________________________
__________________________________________
res2b_branch2a (Conv2D)         (None, 64, 56, 56)   16448
activation_4[0][0]
__________________________________________________________________________________________
__________________________________________
bn2b_branch2a (BatchNormalizati (None, 64, 56, 56)   256
res2b_branch2a[0][0]
__________________________________________________________________________________________
__________________________________________
activation_5 (Activation)       (None, 64, 56, 56)   0
bn2b_branch2a[0][0]
__________________________________________________________________________________________
__________________________________________
res2b_branch2b (Conv2D)         (None, 64, 56, 56)   36928
activation_5[0][0]
__________________________________________________________________________________________
__________________________________________
bn2b_branch2b (BatchNormalizati (None, 64, 56, 56)   256
res2b_branch2b[0][0]
__________________________________________________________________________________________
__________________________________________
activation_6 (Activation)       (None, 64, 56, 56)   0
bn2b_branch2b[0][0]
__________________________________________________________________________________________
__________________________________________
res2b_branch2c (Conv2D)         (None, 256, 56, 56)  16640
activation_6[0][0]
__________________________________________________________________________________________
__________________________________________
bn2b_branch2c (BatchNormalizati (None, 256, 56, 56)  1024
res2b_branch2c[0][0]
__________________________________________________________________________________________
__________________________________________
add_2 (Add)                     (None, 256, 56, 56)  0
bn2b_branch2c[0][0]

activation_4[0][0]
__________________________________________________________________________________________
__________________________________________
activation_7 (Activation)       (None, 256, 56, 56)  0
add_2[0][0]
__________________________________________________________________________________________
```

```
____________________________________________________
res2c_branch2a (Conv2D)          (None, 64, 56, 56)   16448
activation_7[0][0]
____________________________________________________________________________

____________________________________________________
bn2c_branch2a (BatchNormalizati (None, 64, 56, 56)   256
res2c_branch2a[0][0]
____________________________________________________________________________

____________________________________________________
activation_8 (Activation)        (None, 64, 56, 56)   0
bn2c_branch2a[0][0]
____________________________________________________________________________

____________________________________________________
res2c_branch2b (Conv2D)          (None, 64, 56, 56)   36928
activation_8[0][0]
____________________________________________________________________________

____________________________________________________
bn2c_branch2b (BatchNormalizati (None, 64, 56, 56)   256
res2c_branch2b[0][0]
____________________________________________________________________________

____________________________________________________
activation_9 (Activation)        (None, 64, 56, 56)   0
bn2c_branch2b[0][0]
____________________________________________________________________________

____________________________________________________
res2c_branch2c (Conv2D)          (None, 256, 56, 56)  16640
activation_9[0][0]
____________________________________________________________________________

____________________________________________________
bn2c_branch2c (BatchNormalizati (None, 256, 56, 56)  1024
res2c_branch2c[0][0]
____________________________________________________________________________

____________________________________________________
add_3 (Add)                      (None, 256, 56, 56)  0
bn2c_branch2c[0][0]

activation_7[0][0]
____________________________________________________________________________

____________________________________________________
activation_10 (Activation)       (None, 256, 56, 56)  0
add_3[0][0]
____________________________________________________________________________

____________________________________________________
res3a_branch2a (Conv2D)          (None, 128, 28, 28)  32896
activation_10[0][0]
____________________________________________________________________________

____________________________________________________
bn3a_branch2a (BatchNormalizati (None, 128, 28, 28)  512
res3a_branch2a[0][0]
____________________________________________________________________________

____________________________________________________
activation_11 (Activation)       (None, 128, 28, 28)  0
bn3a_branch2a[0][0]
____________________________________________________________________________

____________________________________________________
res3a_branch2b (Conv2D)          (None, 128, 28, 28)  147584
activation_11[0][0]
```

```
__________________________________________________________________________
__________________________________________
bn3a_branch2b (BatchNormalizati (None, 128, 28, 28)  512
res3a_branch2b[0][0]
__________________________________________________________________________
__________________________________________
activation_12 (Activation)      (None, 128, 28, 28)  0
bn3a_branch2b[0][0]
__________________________________________________________________________
__________________________________________
res3a_branch2c (Conv2D)         (None, 512, 28, 28)  66048
activation_12[0][0]
__________________________________________________________________________
__________________________________________
res3a_branch1 (Conv2D)          (None, 512, 28, 28)  131584
activation_10[0][0]
__________________________________________________________________________
__________________________________________
bn3a_branch2c (BatchNormalizati (None, 512, 28, 28)  2048
res3a_branch2c[0][0]
__________________________________________________________________________
__________________________________________
bn3a_branch1 (BatchNormalizatio (None, 512, 28, 28)  2048
res3a_branch1[0][0]
__________________________________________________________________________
__________________________________________
add_4 (Add)                     (None, 512, 28, 28)  0
bn3a_branch2c[0][0]

bn3a_branch1[0][0]
__________________________________________________________________________
__________________________________________
activation_13 (Activation)      (None, 512, 28, 28)  0
add_4[0][0]
__________________________________________________________________________
__________________________________________
res3b_branch2a (Conv2D)         (None, 128, 28, 28)  65664
activation_13[0][0]
__________________________________________________________________________
__________________________________________
bn3b_branch2a (BatchNormalizati (None, 128, 28, 28)  512
res3b_branch2a[0][0]
__________________________________________________________________________
__________________________________________
activation_14 (Activation)      (None, 128, 28, 28)  0
bn3b_branch2a[0][0]
__________________________________________________________________________
__________________________________________
res3b_branch2b (Conv2D)         (None, 128, 28, 28)  147584
activation_14[0][0]
__________________________________________________________________________
__________________________________________
bn3b_branch2b (BatchNormalizati (None, 128, 28, 28)  512
res3b_branch2b[0][0]
__________________________________________________________________________
__________________________________________
activation_15 (Activation)      (None, 128, 28, 28)  0
```

```
bn3b_branch2b[0][0]
__________________________________________________________________________________________________
__________________________________________________
res3b_branch2c (Conv2D)          (None, 512, 28, 28)  66048
activation_15[0][0]
__________________________________________________________________________________________________
__________________________________________________
bn3b_branch2c (BatchNormalizati (None, 512, 28, 28)  2048
res3b_branch2c[0][0]
__________________________________________________________________________________________________
__________________________________________________
add_5 (Add)                      (None, 512, 28, 28)  0
bn3b_branch2c[0][0]

activation_13[0][0]
__________________________________________________________________________________________________
__________________________________________________
activation_16 (Activation)       (None, 512, 28, 28)  0
add_5[0][0]
__________________________________________________________________________________________________
__________________________________________________
res3c_branch2a (Conv2D)          (None, 128, 28, 28)  65664
activation_16[0][0]
__________________________________________________________________________________________________
__________________________________________________
bn3c_branch2a (BatchNormalizati (None, 128, 28, 28)  512
res3c_branch2a[0][0]
__________________________________________________________________________________________________
__________________________________________________
activation_17 (Activation)       (None, 128, 28, 28)  0
bn3c_branch2a[0][0]
__________________________________________________________________________________________________
__________________________________________________
res3c_branch2b (Conv2D)          (None, 128, 28, 28)  147584
activation_17[0][0]
__________________________________________________________________________________________________
__________________________________________________
bn3c_branch2b (BatchNormalizati (None, 128, 28, 28)  512
res3c_branch2b[0][0]
__________________________________________________________________________________________________
__________________________________________________
activation_18 (Activation)       (None, 128, 28, 28)  0
bn3c_branch2b[0][0]
__________________________________________________________________________________________________
__________________________________________________
res3c_branch2c (Conv2D)          (None, 512, 28, 28)  66048
activation_18[0][0]
__________________________________________________________________________________________________
__________________________________________________
bn3c_branch2c (BatchNormalizati (None, 512, 28, 28)  2048
res3c_branch2c[0][0]
__________________________________________________________________________________________________
__________________________________________________
add_6 (Add)                      (None, 512, 28, 28)  0
bn3c_branch2c[0][0]

activation_16[0][0]
```

```
____________________________________________________________________________
____________________
activation_19 (Activation)      (None, 512, 28, 28)  0
add_6[0][0]
____________________________________________________________________________
____________________
res3d_branch2a (Conv2D)         (None, 128, 28, 28)  65664
activation_19[0][0]
____________________________________________________________________________
____________________
bn3d_branch2a (BatchNormalizati (None, 128, 28, 28)  512
res3d_branch2a[0][0]
____________________________________________________________________________
____________________
activation_20 (Activation)      (None, 128, 28, 28)  0
bn3d_branch2a[0][0]
____________________________________________________________________________
____________________
res3d_branch2b (Conv2D)         (None, 128, 28, 28)  147584
activation_20[0][0]
____________________________________________________________________________
____________________
bn3d_branch2b (BatchNormalizati (None, 128, 28, 28)  512
res3d_branch2b[0][0]
____________________________________________________________________________
____________________
activation_21 (Activation)      (None, 128, 28, 28)  0
bn3d_branch2b[0][0]
____________________________________________________________________________
____________________
res3d_branch2c (Conv2D)         (None, 512, 28, 28)  66048
activation_21[0][0]
____________________________________________________________________________
____________________
bn3d_branch2c (BatchNormalizati (None, 512, 28, 28)  2048
res3d_branch2c[0][0]
____________________________________________________________________________
____________________
add_7 (Add)                     (None, 512, 28, 28)  0
bn3d_branch2c[0][0]

activation_19[0][0]
____________________________________________________________________________
____________________
activation_22 (Activation)      (None, 512, 28, 28)  0
add_7[0][0]
____________________________________________________________________________
____________________
res4a_branch2a (Conv2D)         (None, 256, 14, 14)  131328
activation_22[0][0]
____________________________________________________________________________
____________________
bn4a_branch2a (BatchNormalizati (None, 256, 14, 14)  1024
res4a_branch2a[0][0]
____________________________________________________________________________
____________________
activation_23 (Activation)      (None, 256, 14, 14)  0
```

```
bn4a_branch2a[0][0]
__________________________________________________________________________________________________
__________________________________________________
res4a_branch2b (Conv2D)         (None, 256, 14, 14)  590080
activation_23[0][0]
__________________________________________________________________________________________________
__________________________________________________
bn4a_branch2b (BatchNormalizati (None, 256, 14, 14)  1024
res4a_branch2b[0][0]
__________________________________________________________________________________________________
__________________________________________________
activation_24 (Activation)      (None, 256, 14, 14)  0
bn4a_branch2b[0][0]
__________________________________________________________________________________________________
__________________________________________________
res4a_branch2c (Conv2D)         (None, 1024, 14, 14) 263168
activation_24[0][0]
__________________________________________________________________________________________________
__________________________________________________
res4a_branch1 (Conv2D)          (None, 1024, 14, 14) 525312
activation_22[0][0]
__________________________________________________________________________________________________
__________________________________________________
bn4a_branch2c (BatchNormalizati (None, 1024, 14, 14) 4096
res4a_branch2c[0][0]
__________________________________________________________________________________________________
__________________________________________________
bn4a_branch1 (BatchNormalizatio (None, 1024, 14, 14) 4096
res4a_branch1[0][0]
__________________________________________________________________________________________________
__________________________________________________
add_8 (Add)                     (None, 1024, 14, 14) 0
bn4a_branch2c[0][0]

bn4a_branch1[0][0]
__________________________________________________________________________________________________
__________________________________________________
activation_25 (Activation)      (None, 1024, 14, 14) 0
add_8[0][0]
__________________________________________________________________________________________________
__________________________________________________
res4b_branch2a (Conv2D)         (None, 256, 14, 14)  262400
activation_25[0][0]
__________________________________________________________________________________________________
__________________________________________________
bn4b_branch2a (BatchNormalizati (None, 256, 14, 14)  1024
res4b_branch2a[0][0]
__________________________________________________________________________________________________
__________________________________________________
activation_26 (Activation)      (None, 256, 14, 14)  0
bn4b_branch2a[0][0]
__________________________________________________________________________________________________
__________________________________________________
res4b_branch2b (Conv2D)         (None, 256, 14, 14)  590080
activation_26[0][0]
__________________________________________________________________________________________________
__________________________________________________
```

```
bn4b_branch2b (BatchNormalizati (None, 256, 14, 14)  1024
res4b_branch2b[0][0]
__________________________________________________________________________________________
activation_27 (Activation)      (None, 256, 14, 14)  0
bn4b_branch2b[0][0]
__________________________________________________________________________________________
res4b_branch2c (Conv2D)         (None, 1024, 14, 14) 263168
activation_27[0][0]
__________________________________________________________________________________________
bn4b_branch2c (BatchNormalizati (None, 1024, 14, 14) 4096
res4b_branch2c[0][0]
__________________________________________________________________________________________
add_9 (Add)                     (None, 1024, 14, 14) 0
bn4b_branch2c[0][0]

activation_25[0][0]
__________________________________________________________________________________________
activation_28 (Activation)      (None, 1024, 14, 14) 0
add_9[0][0]
__________________________________________________________________________________________
res4c_branch2a (Conv2D)         (None, 256, 14, 14)  262400
activation_28[0][0]
__________________________________________________________________________________________
bn4c_branch2a (BatchNormalizati (None, 256, 14, 14)  1024
res4c_branch2a[0][0]
__________________________________________________________________________________________
activation_29 (Activation)      (None, 256, 14, 14)  0
bn4c_branch2a[0][0]
__________________________________________________________________________________________
res4c_branch2b (Conv2D)         (None, 256, 14, 14)  590080
activation_29[0][0]
__________________________________________________________________________________________
bn4c_branch2b (BatchNormalizati (None, 256, 14, 14)  1024
res4c_branch2b[0][0]
__________________________________________________________________________________________
activation_30 (Activation)      (None, 256, 14, 14)  0
bn4c_branch2b[0][0]
__________________________________________________________________________________________
res4c_branch2c (Conv2D)         (None, 1024, 14, 14) 263168
activation_30[0][0]
__________________________________________________________________________________________
bn4c_branch2c (BatchNormalizati (None, 1024, 14, 14) 4096
res4c_branch2c[0][0]
__________________________________________________________________________________________
```

```
____________________________________________
add_10 (Add)                    (None, 1024, 14, 14) 0
bn4c_branch2c[0][0]

activation_28[0][0]
____________________________________________________________________________
____________________________________________
activation_31 (Activation)      (None, 1024, 14, 14) 0
add_10[0][0]
____________________________________________________________________________
____________________________________________
res4d_branch2a (Conv2D)         (None, 256, 14, 14)  262400
activation_31[0][0]
____________________________________________________________________________
____________________________________________
bn4d_branch2a (BatchNormalizati (None, 256, 14, 14)  1024
res4d_branch2a[0][0]
____________________________________________________________________________
____________________________________________
activation_32 (Activation)      (None, 256, 14, 14)  0
bn4d_branch2a[0][0]
____________________________________________________________________________
____________________________________________
res4d_branch2b (Conv2D)         (None, 256, 14, 14)  590080
activation_32[0][0]
____________________________________________________________________________
____________________________________________
bn4d_branch2b (BatchNormalizati (None, 256, 14, 14)  1024
res4d_branch2b[0][0]
____________________________________________________________________________
____________________________________________
activation_33 (Activation)      (None, 256, 14, 14)  0
bn4d_branch2b[0][0]
____________________________________________________________________________
____________________________________________
res4d_branch2c (Conv2D)         (None, 1024, 14, 14) 263168
activation_33[0][0]
____________________________________________________________________________
____________________________________________
bn4d_branch2c (BatchNormalizati (None, 1024, 14, 14) 4096
res4d_branch2c[0][0]
____________________________________________________________________________
____________________________________________
add_11 (Add)                    (None, 1024, 14, 14) 0
bn4d_branch2c[0][0]

activation_31[0][0]
____________________________________________________________________________
____________________________________________
activation_34 (Activation)      (None, 1024, 14, 14) 0
add_11[0][0]
____________________________________________________________________________
____________________________________________
res4e_branch2a (Conv2D)         (None, 256, 14, 14)  262400
activation_34[0][0]
____________________________________________________________________________
____________________________________________
```

```
bn4e_branch2a (BatchNormalizati (None, 256, 14, 14)  1024
res4e_branch2a[0][0]
__________________________________________________________________________________________
activation_35 (Activation)      (None, 256, 14, 14)  0
bn4e_branch2a[0][0]
__________________________________________________________________________________________
res4e_branch2b (Conv2D)         (None, 256, 14, 14)  590080
activation_35[0][0]
__________________________________________________________________________________________
bn4e_branch2b (BatchNormalizati (None, 256, 14, 14)  1024
res4e_branch2b[0][0]
__________________________________________________________________________________________
activation_36 (Activation)      (None, 256, 14, 14)  0
bn4e_branch2b[0][0]
__________________________________________________________________________________________
res4e_branch2c (Conv2D)         (None, 1024, 14, 14) 263168
activation_36[0][0]
__________________________________________________________________________________________
bn4e_branch2c (BatchNormalizati (None, 1024, 14, 14) 4096
res4e_branch2c[0][0]
__________________________________________________________________________________________
add_12 (Add)                    (None, 1024, 14, 14) 0
bn4e_branch2c[0][0]

activation_34[0][0]
__________________________________________________________________________________________
activation_37 (Activation)      (None, 1024, 14, 14) 0
add_12[0][0]
__________________________________________________________________________________________
res4f_branch2a (Conv2D)         (None, 256, 14, 14)  262400
activation_37[0][0]
__________________________________________________________________________________________
bn4f_branch2a (BatchNormalizati (None, 256, 14, 14)  1024
res4f_branch2a[0][0]
__________________________________________________________________________________________
activation_38 (Activation)      (None, 256, 14, 14)  0
bn4f_branch2a[0][0]
__________________________________________________________________________________________
res4f_branch2b (Conv2D)         (None, 256, 14, 14)  590080
activation_38[0][0]
__________________________________________________________________________________________
bn4f_branch2b (BatchNormalizati (None, 256, 14, 14)  1024
res4f_branch2b[0][0]
__________________________________________________________________________________________
```

```
__________________________________________________
activation_39 (Activation)      (None, 256, 14, 14)  0
bn4f_branch2b[0][0]
__________________________________________________
__________________________________________________
res4f_branch2c (Conv2D)         (None, 1024, 14, 14) 263168
activation_39[0][0]
__________________________________________________
__________________________________________________
bn4f_branch2c (BatchNormalizati (None, 1024, 14, 14) 4096
res4f_branch2c[0][0]
__________________________________________________
__________________________________________________
add_13 (Add)                    (None, 1024, 14, 14) 0
bn4f_branch2c[0][0]

activation_37[0][0]
__________________________________________________
__________________________________________________
activation_40 (Activation)      (None, 1024, 14, 14) 0
add_13[0][0]
__________________________________________________
__________________________________________________
res5a_branch2a (Conv2D)         (None, 512, 7, 7)    524800
activation_40[0][0]
__________________________________________________
__________________________________________________
bn5a_branch2a (BatchNormalizati (None, 512, 7, 7)    2048
res5a_branch2a[0][0]
__________________________________________________
__________________________________________________
activation_41 (Activation)      (None, 512, 7, 7)    0
bn5a_branch2a[0][0]
__________________________________________________
__________________________________________________
res5a_branch2b (Conv2D)         (None, 512, 7, 7)    2359808
activation_41[0][0]
__________________________________________________
__________________________________________________
bn5a_branch2b (BatchNormalizati (None, 512, 7, 7)    2048
res5a_branch2b[0][0]
__________________________________________________
__________________________________________________
activation_42 (Activation)      (None, 512, 7, 7)    0
bn5a_branch2b[0][0]
__________________________________________________
__________________________________________________
res5a_branch2c (Conv2D)         (None, 2048, 7, 7)   1050624
activation_42[0][0]
__________________________________________________
__________________________________________________
res5a_branch1 (Conv2D)          (None, 2048, 7, 7)   2099200
activation_40[0][0]
__________________________________________________
__________________________________________________
bn5a_branch2c (BatchNormalizati (None, 2048, 7, 7)   8192
res5a_branch2c[0][0]
```

```
____________________________________________________________________________
____________________________________________
bn5a_branch1 (BatchNormalizatio (None, 2048, 7, 7)   8192
res5a_branch1[0][0]
____________________________________________________________________________
____________________________________________
add_14 (Add)                    (None, 2048, 7, 7)   0
bn5a_branch2c[0][0]

bn5a_branch1[0][0]
____________________________________________________________________________
____________________________________________
activation_43 (Activation)      (None, 2048, 7, 7)   0
add_14[0][0]
____________________________________________________________________________
____________________________________________
res5b_branch2a (Conv2D)         (None, 512, 7, 7)    1049088
activation_43[0][0]
____________________________________________________________________________
____________________________________________
bn5b_branch2a (BatchNormalizati (None, 512, 7, 7)    2048
res5b_branch2a[0][0]
____________________________________________________________________________
____________________________________________
activation_44 (Activation)      (None, 512, 7, 7)    0
bn5b_branch2a[0][0]
____________________________________________________________________________
____________________________________________
res5b_branch2b (Conv2D)         (None, 512, 7, 7)    2359808
activation_44[0][0]
____________________________________________________________________________
____________________________________________
bn5b_branch2b (BatchNormalizati (None, 512, 7, 7)    2048
res5b_branch2b[0][0]
____________________________________________________________________________
____________________________________________
activation_45 (Activation)      (None, 512, 7, 7)    0
bn5b_branch2b[0][0]
____________________________________________________________________________
____________________________________________
res5b_branch2c (Conv2D)         (None, 2048, 7, 7)   1050624
activation_45[0][0]
____________________________________________________________________________
____________________________________________
bn5b_branch2c (BatchNormalizati (None, 2048, 7, 7)   8192
res5b_branch2c[0][0]
____________________________________________________________________________
____________________________________________
add_15 (Add)                    (None, 2048, 7, 7)   0
bn5b_branch2c[0][0]

activation_43[0][0]
____________________________________________________________________________
____________________________________________
activation_46 (Activation)      (None, 2048, 7, 7)   0
add_15[0][0]
____________________________________________________________________________
```

```
__________________________________________________
res5c_branch2a (Conv2D)         (None, 512, 7, 7)    1049088
activation_46[0][0]
__________________________________________________
__________________________________________________
bn5c_branch2a (BatchNormalizati (None, 512, 7, 7)    2048
res5c_branch2a[0][0]
__________________________________________________
__________________________________________________
activation_47 (Activation)      (None, 512, 7, 7)    0
bn5c_branch2a[0][0]
__________________________________________________
__________________________________________________
res5c_branch2b (Conv2D)         (None, 512, 7, 7)    2359808
activation_47[0][0]
__________________________________________________
__________________________________________________
bn5c_branch2b (BatchNormalizati (None, 512, 7, 7)    2048
res5c_branch2b[0][0]
__________________________________________________
__________________________________________________
activation_48 (Activation)      (None, 512, 7, 7)    0
bn5c_branch2b[0][0]
__________________________________________________
__________________________________________________
res5c_branch2c (Conv2D)         (None, 2048, 7, 7)   1050624
activation_48[0][0]
__________________________________________________
__________________________________________________
bn5c_branch2c (BatchNormalizati (None, 2048, 7, 7)   8192
res5c_branch2c[0][0]
__________________________________________________
__________________________________________________
add_16 (Add)                    (None, 2048, 7, 7)   0
bn5c_branch2c[0][0]

activation_46[0][0]
__________________________________________________
__________________________________________________
activation_49 (Activation)      (None, 2048, 7, 7)   0
add_16[0][0]
__________________________________________________
__________________________________________________
avg_pool (GlobalAveragePooling2 (None, 2048)         0
activation_49[0][0]
__________________________________________________
__________________________________________________
fc1000 (Dense)                  (None, 1000)         2049000
avg_pool[0][0]
==================================================
==================================
Total params: 25,636,712
Trainable params: 25,583,592
Non-trainable params: 53,120
__________________________________________________
__________________________________________________
```

- Look at the 1000 classes

In [54]:
```
tmp = resnet.decode_predictions(eye(1000), top=1)  # decode th
e 1000-dim class vector
tmp2 = [i[0][1] for i in tmp]
tmp2.sort()
print(" ".join(tmp2))
```

```
Afghan_hound African_chameleon African_crocodile African_elep
hant African_grey African_hunting_dog Airedale American_Staff
ordshire_terrier American_alligator American_black_bear Ameri
can_chameleon American_coot American_egret American_lobster A
ngora Appenzeller Arabian_camel Arctic_fox Australian_terrier
Band_Aid Bedlington_terrier Bernese_mountain_dog Blenheim_spa
niel Border_collie Border_terrier Boston_bull Bouvier_des_Fla
ndres Brabancon_griffon Brittany_spaniel CD_player Cardigan C
hesapeake_Bay_retriever Chihuahua Christmas_stocking Crock_Po
t Dandie_Dinmont Doberman Dungeness_crab Dutch_oven Egyptian_
cat English_foxhound English_setter English_springer EntleBuc
her Eskimo_dog European_fire_salamander European_gallinule Fr
ench_bulldog French_horn French_loaf German_shepherd German_s
hort-haired_pointer Gila_monster Gordon_setter Granny_Smith G
reat_Dane Great_Pyrenees Greater_Swiss_Mountain_dog Ibizan_ho
und Indian_cobra Indian_elephant Irish_setter Irish_terrier I
rish_water_spaniel Irish_wolfhound Italian_greyhound Japanese
_spaniel Kerry_blue_terrier Komodo_dragon Labrador_retriever
Lakeland_terrier Leonberg Lhasa Loafer Madagascar_cat Maltese
_dog Mexican_hairless Model_T Newfoundland Norfolk_terrier No
rwegian_elkhound Norwich_terrier Old_English_sheepdog Pekines
e Pembroke Persian_cat Petri_dish Polaroid_camera Pomeranian
Rhodesian_ridgeback Rottweiler Saint_Bernard Saluki Samoyed S
cotch_terrier Scottish_deerhound Sealyham_terrier Shetland_sh
eepdog Shih-Tzu Siamese_cat Siberian_husky Staffordshire_bull
terrier Sussex_spaniel Tibetan_mastiff Tibetan_terrier Walker
_hound Weimaraner Welsh_springer_spaniel West_Highland_white_
terrier Windsor_tie Yorkshire_terrier abacus abaya academic_g
own accordion acorn acorn_squash acoustic_guitar admiral affe
npinscher agama agaric aircraft_carrier airliner airship alba
tross alligator_lizard alp altar ambulance amphibian analog_c
lock anemone_fish ant apiary apron armadillo artichoke ashcan
assault_rifle axolotl baboon backpack badger bagel bakery bal
ance_beam bald_eagle balloon ballplayer ballpoint banana band
ed_gecko banjo bannister barbell barber_chair barbershop barn
barn_spider barometer barracouta barrel barrow baseball basen
ji basketball basset bassinet bassoon bath_towel bathing_cap
bathtub beach_wagon beacon beagle beaker bearskin beaver bee
bee_eater beer_bottle beer_glass bell_cote bell_pepper bib bi
cycle-built-for-two bighorn bikini binder binoculars birdhous
e bison bittern black-and-tan_coonhound black-footed_ferret b
lack_and_gold_garden_spider black_grouse black_stork black_sw
an black_widow bloodhound bluetick boa_constrictor boathouse
bobsled bolete bolo_tie bonnet book_jacket bookcase bookshop
borzoi bottlecap bow bow_tie box_turtle boxer brain_coral bra
mbling brass brassiere breakwater breastplate briard broccoli
```

broom brown_bear bubble bucket buckeye buckle bulbul bull_mas
tiff bullet_train bulletproof_vest bullfrog burrito bustard b
utcher_shop butternut_squash cab cabbage_butterfly cairn cald
ron can_opener candle cannon canoe capuchin car_mirror car_wh
eel carbonara cardigan cardoon carousel carpenter's_kit carto
n cash_machine cassette cassette_player castle catamaran caul
iflower cello cellular_telephone centipede chain chain_mail c
hain_saw chainlink_fence chambered_nautilus cheeseburger chee
tah chest chickadee chiffonier chime chimpanzee china_cabinet
chiton chocolate_sauce chow church cicada cinema cleaver clif
f cliff_dwelling cloak clog clumber cock cocker_spaniel cockr
oach cocktail_shaker coffee_mug coffeepot coho coil collie co
lobus combination_lock comic_book common_iguana common_newt c
omputer_keyboard conch confectionery consomme container_ship
convertible coral_fungus coral_reef corkscrew corn cornet cou
cal cougar cowboy_boot cowboy_hat coyote cradle crane crane c
rash_helmet crate crayfish crib cricket croquet_ball crosswor
d_puzzle crutch cucumber cuirass cup curly-coated_retriever c
ustard_apple daisy dalmatian dam damselfly desk desktop_compu
ter dhole dial_telephone diamondback diaper digital_clock dig
ital_watch dingo dining_table dishrag dishwasher disk_brake d
ock dogsled dome doormat dough dowitcher dragonfly drake dril
ling_platform drum drumstick dugong dumbbell dung_beetle ear
earthstar echidna eel eft eggnog electric_fan electric_guitar
electric_locomotive electric_ray entertainment_center envelop
e espresso espresso_maker face_powder feather_boa fiddler_cra
b fig file fire_engine fire_screen fireboat flagpole flamingo
flat-coated_retriever flatworm flute fly folding_chair footba
ll_helmet forklift fountain fountain_pen four-poster fox_squi
rrel freight_car frilled_lizard frying_pan fur_coat gar garba
ge_truck garden_spider garter_snake gas_pump gasmask gazelle
geyser giant_panda giant_schnauzer gibbon go-kart goblet gold
en_retriever goldfinch goldfish golf_ball golfcart gondola go
ng goose gorilla gown grand_piano grasshopper great_grey_owl
great_white_shark green_lizard green_mamba green_snake greenh
ouse grey_fox grey_whale grille grocery_store groenendael gro
om ground_beetle guacamole guenon guillotine guinea_pig gyrom
itra hair_slide hair_spray half_track hammer hammerhead hampe
r hamster hand-held_computer hand_blower handkerchief hard_di
sc hare harmonica harp hartebeest harvester harvestman hatche
t hay head_cabbage hen hen-of-the-woods hermit_crab hip hippo
potamus hog hognose_snake holster home_theater honeycomb hook
hoopskirt horizontal_bar hornbill horned_viper horse_cart hot
_pot hotdog hourglass house_finch howler_monkey hummingbird h
yena iPod ibex ice_bear ice_cream ice_lolly impala indigo_bun
ting indri iron isopod jacamar jack-o'-lantern jackfruit jagu
ar jay jean jeep jellyfish jersey jigsaw_puzzle jinrikisha jo
ystick junco keeshond kelpie killer_whale kimono king_crab ki
ng_penguin king_snake kit_fox kite knee_pad knot koala komond
or kuvasz lab_coat lacewing ladle ladybug lakeside lampshade
langur laptop lawn_mower leaf_beetle leafhopper leatherback_t
urtle lemon lens_cap leopard lesser_panda letter_opener libra
ry lifeboat lighter limousine limpkin liner lion lionfish lip
stick little_blue_heron llama loggerhead long-horned_beetle l
orikeet lotion loudspeaker loupe lumbermill lycaenid lynx mac
aque macaw magnetic_compass magpie mailbag mailbox maillot ma
illot malamute malinois manhole_cover mantis maraca marimba m

armoset marmot mashed_potato mask matchstick maypole maze mea
suring_cup meat_loaf medicine_chest meerkat megalith menu mic
rophone microwave military_uniform milk_can miniature_pinsche
r miniature_poodle miniature_schnauzer minibus miniskirt mini
van mink missile mitten mixing_bowl mobile_home modem monarch
monastery mongoose monitor moped mortar mortarboard mosque mo
squito_net motor_scooter mountain_bike mountain_tent mouse mo
usetrap moving_van mud_turtle mushroom muzzle nail neck_brace
necklace nematode night_snake nipple notebook obelisk oboe oc
arina odometer oil_filter orange orangutan organ oscilloscope
ostrich otter otterhound overskirt ox oxcart oxygen_mask oyst
ercatcher packet paddle paddlewheel padlock paintbrush pajama
palace panpipe paper_towel papillon parachute parallel_bars p
ark_bench parking_meter partridge passenger_car patas patio p
ay-phone peacock pedestal pelican pencil_box pencil_sharpener
perfume photocopier pick pickelhaube picket_fence pickup pier
piggy_bank pill_bottle pillow pineapple ping-pong_ball pinwhe
el pirate pitcher pizza plane planetarium plastic_bag plate p
late_rack platypus plow plunger pole polecat police_van pomeg
ranate poncho pool_table pop_bottle porcupine pot potpie pott
er's_wheel power_drill prairie_chicken prayer_rug pretzel pri
nter prison proboscis_monkey projectile projector promontory
ptarmigan puck puffer pug punching_bag purse quail quill quil
t racer racket radiator radio radio_telescope rain_barrel ram
rapeseed recreational_vehicle red-backed_sandpiper red-breast
ed_merganser red_fox red_wine red_wolf redbone redshank reel
reflex_camera refrigerator remote_control restaurant revolver
rhinoceros_beetle rifle ringlet ringneck_snake robin rock_bea
uty rock_crab rock_python rocking_chair rotisserie rubber_era
ser ruddy_turnstone ruffed_grouse rugby_ball rule running_sho
e safe safety_pin saltshaker sandal sandbar sarong sax scabba
rd scale schipperke school_bus schooner scoreboard scorpion s
creen screw screwdriver scuba_diver sea_anemone sea_cucumber
sea_lion sea_slug sea_snake sea_urchin seashore seat_belt sew
ing_machine shield shoe_shop shoji shopping_basket shopping_c
art shovel shower_cap shower_curtain siamang sidewinder silky
_terrier ski ski_mask skunk sleeping_bag slide_rule sliding_d
oor slot sloth_bear slug snail snorkel snow_leopard snowmobil
e snowplow soap_dispenser soccer_ball sock soft-coated_wheate
n_terrier solar_dish sombrero sorrel soup_bowl space_bar spac
e_heater space_shuttle spaghetti_squash spatula speedboat spi
der_monkey spider_web spindle spiny_lobster spoonbill sports_
car spotlight spotted_salamander squirrel_monkey stage standa
rd_poodle standard_schnauzer starfish steam_locomotive steel_
arch_bridge steel_drum stethoscope stingray stinkhorn stole s
tone_wall stopwatch stove strainer strawberry street_sign str
eetcar stretcher studio_couch stupa sturgeon submarine suit s
ulphur-crested_cockatoo sulphur_butterfly sundial sunglass su
nglasses sunscreen suspension_bridge swab sweatshirt swimming
_trunks swing switch syringe tabby table_lamp tailed_frog tan
k tape_player tarantula teapot teddy television tench tennis_
ball terrapin thatch theater_curtain thimble three-toed_sloth
thresher throne thunder_snake tick tiger tiger_beetle tiger_c
at tiger_shark tile_roof timber_wolf titi toaster tobacco_sho
p toilet_seat toilet_tissue torch totem_pole toucan tow_truck
toy_poodle toy_terrier toyshop tractor traffic_light trailer_
truck tray tree_frog trench_coat triceratops tricycle trifle

```
trilobite trimaran tripod triumphal_arch trolleybus trombone
tub turnstile tusker typewriter_keyboard umbrella unicycle up
right vacuum valley vase vault velvet vending_machine vestmen
t viaduct vine_snake violin vizsla volcano volleyball vulture
waffle_iron walking_stick wall_clock wallaby wallet wardrobe
warplane warthog washbasin washer water_bottle water_buffalo
water_jug water_ouzel water_snake water_tower weasel web_site
weevil whippet whiptail whiskey_jug whistle white_stork white
_wolf wig wild_boar window_screen window_shade wine_bottle wi
ng wire-haired_fox_terrier wok wolf_spider wombat wood_rabbit
wooden_spoon wool worm_fence wreck yawl yellow_lady's_slipper
yurt zebra zucchini
```

- Predict on the loaded image

In [61]:
```
# make a prediction on the image
preds = model.predict(xi)

# decode the results into a list of tuples (class, description
, probability)
predsc = resnet.decode_predictions(preds, top=3)[0]

for i in predsc:
    print(i)
```

```
('n02504013', 'Indian_elephant', 0.8574033)
('n01871265', 'tusker', 0.13916093)
('n02504458', 'African_elephant', 0.003434056)
```

Pre-trained Image Features

- The network is trained on ImageNet, which contains many classes and images.
- The learned features generalize well to other tasks
  - capture high-level discriminative information about objects

In [8]:
```
# load dessert dataset
import glob

imglist = glob.glob('imgs/dessert/*/*.jpg')
Xraw = []
Xim = zeros((len(imglist), 3, 224, 224))
Ylab = []
for i,img_path in enumerate(imglist):
    img = image.load_img(img_path, target_size=(224, 224))
    Xraw.append(img)
    x = image.img_to_array(img)
    x = expand_dims(x, axis=0)
    x = resnet.preprocess_input(x)
    Xim[i,:] = x
    Ylab.append(img_path.split('/')[2])
```

In [9]:
```python
print(Xim.shape)
print(len(Ylab))
```

```
(200, 3, 224, 224)
200
```

In [67]:
```python
# convert class strings into integers
print("class labels (strings):", unique(Ylab))
le = preprocessing.LabelEncoder()
Y = le.fit_transform(Ylab)
print("Converted labels:")
print(Y)
```

```
class labels (strings): ['frozen_yogurt' 'ice_cream']
Converted labels:
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
```

- Show a few examples of "ice cream" and "frozen yogurt"

In [69]:
```python
inds = [0, 1, 2, 3, 4, -1, -2, -3, -4, -5]
tmp = [Xraw[i] for i in inds]
t =  [Ylab[i] for i in inds]
plt.figure(figsize=(9,4))
show_imgs(tmp, nc=5, titles=t)
```

Computing Image Features

- Use the pre-trained network as a feature extractor
  - remove the last layer (the classifier)
  - apply average pooling on the feature map
    - 7x7x2048 --> 1x2048

In [12]:
```
# create an instance of the model w/o the last layer
model_f = resnet.ResNet50(weights='imagenet', include_top=Fals
e, pooling='avg')

# compute the features
Xf = model_f.predict(Xim)
print(Xf.shape)
```

```
(200, 2048)
```

- Now use the features to train a binary classifier
- First setup the dataset

In [80]:
```
# convert class labels to binary indicators
Yb = keras.utils.np_utils.to_categorical(Y)

# randomly split data into train and test set
( trainXf, testXf,        # features
  trainY, testY,          # class labels
  trainXim, testXim,      # pre-processed images
  trainYb, testYb ,       # binary class vector (for Keras)
  trainXraw, testXraw     # original images
) = \
  model_selection.train_test_split(Xf, Y, Xim, Yb, Xraw,
  train_size=0.8, test_size=0.2, random_state=4487)

trainX = trainXim.reshape((trainXim.shape[0], -1))
testX  = testXim.reshape((testXim.shape[0], -1))

print(trainXf.shape)
print(testXf.shape)
print(trainXim.shape)
print(trainX.shape)
```

```
(160, 2048)
(40, 2048)
(160, 3, 224, 224)
(160, 150528)
```

- Setup validation set for later

In [14]:
```
# generate a fixed validation set using 10% of the training se
t
vtrainXim, validXim, vtrainYb, validYb = \
  model_selection.train_test_split(trainXim, trainYb,
  train_size=0.9, test_size=0.1, random_state=6487)

# validation data
validset  = (validXim, validYb)
```

- Use the image features to train a standard SVM classifier

In [70]:
```
# setup the list of parameters to try
paramgrid = {'C': logspace(-3,3,13)}
print(paramgrid)

# setup cross-validation
svmcv = model_selection.GridSearchCV(svm.SVC(kernel='linear'),
paramgrid, cv=5,
                                     n_jobs=-1, verbose=True)

# run cross-validation (train for each split)
svmcv.fit(trainXf, trainY);
```

```
{'C': array([1.00000000e-03, 3.16227766e-03, 1.00000000e-02,
3.16227766e-02,
       1.00000000e-01, 3.16227766e-01, 1.00000000e+00, 3.1622
7766e+00,
       1.00000000e+01, 3.16227766e+01, 1.00000000e+02, 3.1622
7766e+02,
       1.00000000e+03])}
Fitting 5 folds for each of 13 candidates, totalling 65 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 conc
urrent workers.
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:    2.7s
[Parallel(n_jobs=-1)]: Done  65 out of  65 | elapsed:    3.1s
finished
```

- Predict with the classifier
- The accuracy is great!

In [16]:
```
# predict
predY = svmcv.predict(testXf)

acc = metrics.accuracy_score(testY, predY)
print("test accuracy = " + str(acc))
```

```
test accuracy = 0.925
```

- For comparison, also train an SVM on the raw images
- The accuracy is really bad (almost chance level)

In [17]:
```
svmcv_im = model_selection.GridSearchCV(svm.SVC(kernel='linear
'), paramgrid, cv=5,
                                        n_jobs=-1, verbose=True)

# run cross-validation (train for each split)
svmcv_im.fit(trainX, trainY);


# Directly use svmcv to make predictions
predY = svmcv_im.predict(testX)

acc = metrics.accuracy_score(testY, predY)
print("test accuracy = " + str(acc))
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 12 conc
urrent workers.

Fitting 5 folds for each of 13 candidates, totalling 65 fits

/anaconda3/lib/python3.6/site-packages/sklearn/externals/jobl
ib/externals/loky/process_executor.py:700: UserWarning: A wor
ker stopped while some jobs were given to the executor. This
can be caused by a too short worker timeout or by a memory le
ak.
  "timeout or by a memory leak.", UserWarning
[Parallel(n_jobs=-1)]: Done  26 tasks      | elapsed:  1.5min
[Parallel(n_jobs=-1)]: Done  65 out of  65 | elapsed:  2.7min
finished

test accuracy = 0.525
```

Transfer learning and fine-tuning

- The network can also be "fine-tuned" for a new image classification task.
  - This is called transfer learning.
- Rather than retrain the whole network,
  - fix the lower layers that extract features (no need to train them since they are good features)
  - train the last few layers to perform the new task
- Does not need as much data, compared to the original ImageNet.

- First, let's see ResNet50 predicts on our images

In [27]:
```
# try the original ResNet50
testYscore = model.predict(testXim)
```

In [84]:
```
# get the class labels
predY = resnet.decode_predictions(testYscore, top=1)
testYcl = le.inverse_transform(testY)

# make titles
titles = []
for i in range(len(predY)):
    titles.append(predY[i][0][1] + "\n(" + testYcl[i] + ")")

# make a plot: predicted (true)
plt.figure(figsize=(9,20))
show_imgs(testXraw, nc=5, titles=titles)
```

- We will use the `Model` class
  - allows for more complex layer interactions than "Sequential"
- Connect layers one-by-one
  - instantiate the layer (with hyperparameters)
  - then pass the previous layer to it.

In [20]:
```
#random.seed(4487); tensorflow.set_random_seed(4487)

# create the base pre-trained model with-out the classifier
base_model = resnet.ResNet50(weights='imagenet', include_top=F
alse)

# start with the output of the ResNet50 (7x7x2048)
x = base_model.output

# for each channel, average all the features (1x2048)
x = GlobalAveragePooling2D()(x)

# fully-connected layer (1 x32)
# (only two classes so don't need so many)
x = Dense(32, activation='relu')(x)

# finally, the softmax for the classifier (2 classes)
predictions = Dense(2, activation='softmax')(x)
```

- Create the Model
  - specify the input and output layers
  - the network is everything in between
- Fix the weights of the layers of ResNet so they will not change during training

In [ ]:
```
# build the model for training
# - need to specify the input layer and the output layer
model_ft = Model(inputs=base_model.input, outputs=predictions)

# fix the layers of the ResNet50.
for layer in base_model.layers:
    layer.trainable = False

# compile the model - only the layers that we added will be tr
ained
model_ft.compile(optimizer='rmsprop',
                 loss='categorical_crossentropy', metrics=['accur
acy'])
```

- Setup the data augmentation generator

In [21]:
```
from keras.preprocessing.image import ImageDataGenerator

def add_gauss_noise(X, sigma2=0.05):
    # add Gaussian noise with zero mean, and variance sigma2
    return X + random.normal(0, sigma2, X.shape)

# build the data augmenter
datagen = ImageDataGenerator(
    rotation_range=10,          # image rotation
    width_shift_range=0.2,      # image shifting
    height_shift_range=0.2,     # image shifting
    shear_range=0.1,            # shear transformation
    zoom_range=0.1,             # zooming
    horizontal_flip=True,
    preprocessing_function=add_gauss_noise,
)

# fit (required for some normalization augmentations)
datagen.fit(vtrainXim)
```

- Train the model
  - only the last layers that we added are updated

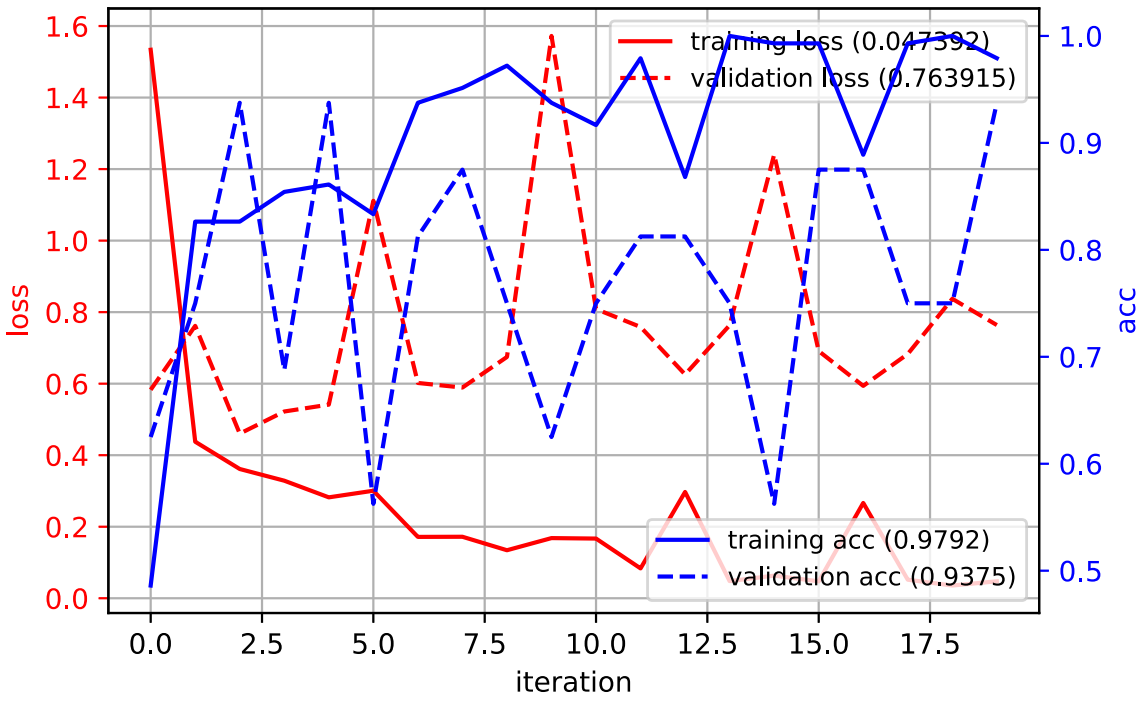
In [22]:
```
# train the model on the new data for a few epochs
bsize = 32
callbacks_list = []
history = model_ft.fit_generator(
            datagen.flow(vtrainXim, vtrainYb, batch_size=bsize
),  # data from generator
            steps_per_epoch=len(vtrainXim)/bsize,    # should
be number of batches per epoch
            epochs=20,
            callbacks=callbacks_list,
            validation_data=validset, verbose=True)
```

```
Epoch 1/20
5/4 [==============================] - 102s 20s/step - los
s: 1.4617 - acc: 0.4925 - val_loss: 0.5829 - val_acc: 0.6250
Epoch 2/20
5/4 [==============================] - 87s 17s/step - loss
: 0.4334 - acc: 0.8252 - val_loss: 0.7617 - val_acc: 0.7500
Epoch 3/20
5/4 [==============================] - 87s 17s/step - loss
: 0.3738 - acc: 0.8200 - val_loss: 0.4603 - val_acc: 0.9375
Epoch 4/20
5/4 [==============================] - 87s 17s/step - loss
: 0.3278 - acc: 0.8559 - val_loss: 0.5222 - val_acc: 0.6875
Epoch 5/20
5/4 [==============================] - 87s 17s/step - loss
: 0.2908 - acc: 0.8571 - val_loss: 0.5414 - val_acc: 0.9375
Epoch 6/20
5/4 [==============================] - 87s 17s/step - loss
: 0.3211 - acc: 0.8160 - val_loss: 1.1114 - val_acc: 0.5625
Epoch 7/20
5/4 [==============================] - 87s 17s/step - loss
: 0.1661 - acc: 0.9375 - val_loss: 0.6020 - val_acc: 0.8125
Epoch 8/20
5/4 [==============================] - 87s 17s/step - loss
: 0.1656 - acc: 0.9554 - val_loss: 0.5891 - val_acc: 0.8750
Epoch 9/20
5/4 [==============================] - 87s 17s/step - loss
: 0.1315 - acc: 0.9745 - val_loss: 0.6747 - val_acc: 0.7500
Epoch 10/20
5/4 [==============================] - 87s 17s/step - loss
: 0.1815 - acc: 0.9323 - val_loss: 1.5723 - val_acc: 0.6250
Epoch 11/20
5/4 [==============================] - 87s 17s/step - loss
: 0.1607 - acc: 0.9184 - val_loss: 0.8087 - val_acc: 0.7500
Epoch 12/20
5/4 [==============================] - 87s 17s/step - loss
: 0.0803 - acc: 0.9809 - val_loss: 0.7585 - val_acc: 0.8125
Epoch 13/20
5/4 [==============================] - 87s 17s/step - loss
: 0.2739 - acc: 0.8791 - val_loss: 0.6258 - val_acc: 0.8125
Epoch 14/20
5/4 [==============================] - 87s 17s/step - loss
: 0.0477 - acc: 1.0000 - val_loss: 0.7634 - val_acc: 0.7500
Epoch 15/20
5/4 [==============================] - 87s 17s/step - loss
```

```
: 0.0636 - acc: 0.9936 - val_loss: 1.2406 - val_acc: 0.5625
Epoch 16/20
5/4 [==============================] - 87s 17s/step - loss
: 0.0503 - acc: 0.9884 - val_loss: 0.6912 - val_acc: 0.8750
Epoch 17/20
5/4 [==============================] - 87s 17s/step - loss
: 0.3316 - acc: 0.8617 - val_loss: 0.5938 - val_acc: 0.8750
Epoch 18/20
5/4 [==============================] - 87s 17s/step - loss
: 0.0509 - acc: 0.9936 - val_loss: 0.6830 - val_acc: 0.7500
Epoch 19/20
5/4 [==============================] - 87s 17s/step - loss
: 0.0335 - acc: 1.0000 - val_loss: 0.8373 - val_acc: 0.7500
Epoch 20/20
5/4 [==============================] - 88s 18s/step - loss
: 0.0570 - acc: 0.9705 - val_loss: 0.7639 - val_acc: 0.9375
```

- Training/validation curves

In [23]:
```
plot_history(history)
```



- Compute test accuracy

In [89]:
```
predYscore = model_ft.predict(testXim, verbose=False)
predY = argmax(predYscore, axis=1)
acc = metrics.accuracy_score(testY, predY)
print("test accuracy:", acc)
```

```
test accuracy: 0.9
```

- Visualize the predictions

In [98]:
```
# get the class labels
predYcl = le.inverse_transform(predY)
# make titles
titles = []
for i in range(len(predYcl)):
    titles.append(predYcl[i] + "\n(" + testYcl[i] + ")")

# make a plot: predicted (true)
plt.figure(figsize=(9,20))
show_imgs(testXraw, nc=5, titles=titles)
```


(ice_cream)


(ice_cream)


(frozen_yogurt)


(ice_cream)


(ice_cream)

ice_cream


(ice_cream)

ice_cream


(ice_cream)

frozen_yogurt


(frozen_yogurt)

ice_cream


(ice_cream)

frozen_yogurt


(frozen_yogurt)

frozen_yogurt


(ice_cream)

ice_cream


(ice_cream)

frozen_yogurt


(frozen_yogurt)

frozen_yogurt


(frozen_yogurt)

frozen_yogurt


(ice_cream)