dog_app

June 5, 2019

```
In [1]: pip install keras
```

```
Requirement already satisfied: keras in /opt/anaconda3/lib/python3.7/site-packages (2.2.4)
Requirement already satisfied: numpy>=1.9.1 in /opt/anaconda3/lib/python3.7/site-packages (fro
Requirement already satisfied: scipy>=0.14 in /opt/anaconda3/lib/python3.7/site-packages (from
Requirement already satisfied: pyyaml in /opt/anaconda3/lib/python3.7/site-packages (from kera
Requirement already satisfied: six>=1.9.0 in /opt/anaconda3/lib/python3.7/site-packages (from
Requirement already satisfied: keras-applications>=1.0.6 in /opt/anaconda3/lib/python3.7/site-
Requirement already satisfied: keras-preprocessing>=1.0.5 in /opt/anaconda3/lib/python3.7/site
Requirement already satisfied: h5py in /opt/anaconda3/lib/python3.7/site-packages (from keras)
Note: you may need to restart the kernel to use updated packages.
```

```
In [2]: pip install tensorflow
```

```
Requirement already satisfied: tensorflow in /opt/anaconda3/lib/python3.7/site-packages (1.13.
Requirement already satisfied: grpcio>=1.8.6 in /opt/anaconda3/lib/python3.7/site-packages (fr
Requirement already satisfied: wheel>=0.26 in /opt/anaconda3/lib/python3.7/site-packages (from
Requirement already satisfied: termcolor>=1.1.0 in /opt/anaconda3/lib/python3.7/site-packages
Requirement already satisfied: tensorflow-estimator<1.14.0rc0,>=1.13.0 in /opt/anaconda3/lib/p
Requirement already satisfied: keras-preprocessing>=1.0.5 in /opt/anaconda3/lib/python3.7/site
Requirement already satisfied: keras-applications>=1.0.6 in /opt/anaconda3/lib/python3.7/site-
Requirement already satisfied: numpy>=1.13.3 in /opt/anaconda3/lib/python3.7/site-packages (fr
Requirement already satisfied: protobuf>=3.6.1 in /opt/anaconda3/lib/python3.7/site-packages (
Requirement already satisfied: astor>=0.6.0 in /opt/anaconda3/lib/python3.7/site-packages (fro
Requirement already satisfied: absl-py>=0.1.6 in /opt/anaconda3/lib/python3.7/site-packages (f
Requirement already satisfied: six>=1.10.0 in /opt/anaconda3/lib/python3.7/site-packages (from
Requirement already satisfied: tensorboard<1.14.0,>=1.13.0 in /opt/anaconda3/lib/python3.7/sit
Requirement already satisfied: gast>=0.2.0 in /opt/anaconda3/lib/python3.7/site-packages (from
Requirement already satisfied: mock>=2.0.0 in /opt/anaconda3/lib/python3.7/site-packages (from
Requirement already satisfied: h5py in /opt/anaconda3/lib/python3.7/site-packages (from keras-
Requirement already satisfied: setuptools in /opt/anaconda3/lib/python3.7/site-packages (from
Requirement already satisfied: markdown>=2.6.8 in /opt/anaconda3/lib/python3.7/site-packages (
Requirement already satisfied: werkzeug>=0.11.15 in /opt/anaconda3/lib/python3.7/site-packages
Requirement already satisfied: pbr>=0.11 in /opt/anaconda3/lib/python3.7/site-packages (from m
Note: you may need to restart the kernel to use updated packages.
```

```
In [3]: from sklearn.datasets import load_files
        from keras.utils import np_utils
        import numpy as np
        from glob import glob

        def load_dataset(path):
            data = load_files(path)
            dog_files = np.array(data['filenames'])
            dog_targets = np_utils.to_categorical(np.array(data['target']), 133)
            return dog_files, dog_targets

        train_files, train_targets = load_dataset('dogImages/train')
        valid_files, valid_targets = load_dataset('dogImages/valid')
        test_files, test_targets = load_dataset('dogImages/test')

        dog_names = [item[20:-1] for item in sorted(glob("dogImages/train/*/"))]

        print('There are %d total dog categories.' % len(dog_names))
        print('There are %s total dog images.\n' % len(np.hstack([train_files, valid_files, te
        print('There are %d training dog images.' % len(train_files))
        print('There are %d validation dog images.' % len(valid_files))
        print('There are %d test dog images.'% len(test_files))
```

```
Using TensorFlow backend.


There are 133 total dog categories.
There are 8351 total dog images.

There are 6680 training dog images.
There are 835 validation dog images.
There are 836 test dog images.
```

```
In [4]: from keras.applications.resnet50 import ResNet50

        ResNet50_model = ResNet50(weights='imagenet')
```

```
WARNING:tensorflow:From /opt/anaconda3/lib/python3.7/site-packages/tensorflow/python/framework/
Instructions for updating:
Colocations handled automatically by placer.
```

```
In [5]: from keras.preprocessing import image
        from tqdm import tqdm
        from keras.applications.resnet50 import preprocess_input, decode_predictions

        def path_to_tensor(img_path):
            # loads RGB image as PIL.Image.Image type
            img = image.load_img(img_path, target_size=(224, 224))

            x = image.img_to_array(img)
            return np.expand_dims(x, axis=0)


        def paths_to_tensor(img_paths):
            list_of_tensors = [path_to_tensor(img_path) for img_path in tqdm(img_paths)]
            return np.vstack(list_of_tensors)
```

```
In [8]: from PIL import ImageFile
        ImageFile.LOAD_TRUNCATED_IMAGES = True
        from time import time

        train_tensors = paths_to_tensor(train_files).astype('float32')/255
        valid_tensors = paths_to_tensor(valid_files).astype('float32')/255
        test_tensors = paths_to_tensor(test_files).astype('float32')/255
```

```
100%|| 6680/6680 [01:04<00:00, 104.36it/s]
100%|| 835/835 [00:07<00:00, 117.72it/s]
100%|| 836/836 [00:07<00:00, 117.43it/s]
```

```
In [9]: from keras.layers import Conv2D, MaxPooling2D, GlobalAveragePooling2D
        from keras.layers import Dropout, Flatten, Dense
        from keras.models import Sequential
        from keras.layers.advanced_activations import ELU
        from keras.layers.normalization import BatchNormalization

        model = Sequential()

        model.add(Conv2D(filters=16,
                         kernel_size=2,
                         strides=1,
                         padding="same",
                         input_shape=(224, 224, 3)))

        model.add(MaxPooling2D(pool_size=2))

        model.add(Conv2D(filters=32,
                         kernel_size=2,
                         strides=1,
                         padding="same"))

        model.add(MaxPooling2D(pool_size=2))

        model.add(Conv2D(filters=64,
                         kernel_size=2,
                        strides=1,
                        padding="same"))

        model.add(MaxPooling2D(pool_size=2))

        model.add(Conv2D(filters=64,
                        kernel_size=2,
                        strides=1,
                        padding="same"))

        model.add(MaxPooling2D(pool_size=2))

        model.add(Conv2D(filters=64,
                        kernel_size=2,
                        strides=1,
                        padding="same"))

        model.add(GlobalAveragePooling2D())

        model.add(Dense(64, activation="relu"))

        model.add(Dense(133, activation="softmax"))

        model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 224, 224, 16)      208
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 112, 112, 16)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 112, 112, 32)      2080
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 56, 56, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 56, 56, 64)        8256
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 28, 28, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 28, 28, 64)        16448
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 14, 14, 64)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 14, 14, 64)        16448
_________________________________________________________________
global_average_pooling2d_1 ( (None, 64)                0
_________________________________________________________________
```

```
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
dense_2 (Dense)              (None, 133)               8645
=================================================================
Total params: 56,245
Trainable params: 56,245
Non-trainable params: 0
_________________________________________________________________
```

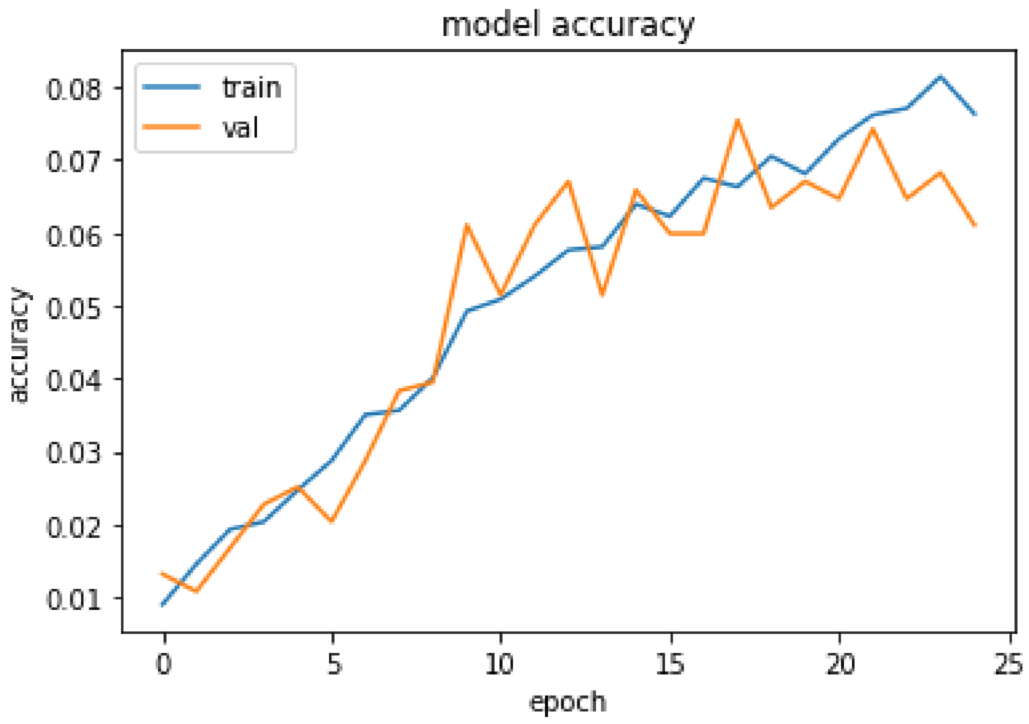
```
In [10]: model.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['accurac
```

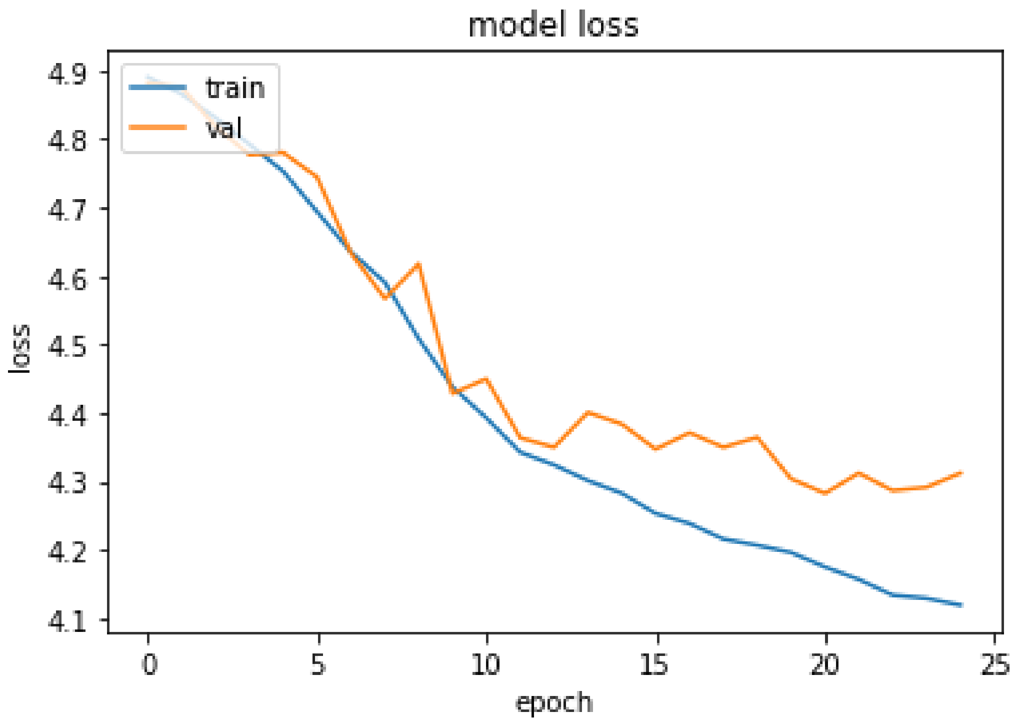
```
In [11]: from keras.callbacks import ModelCheckpoint
         from keras.preprocessing.image import ImageDataGenerator
         import matplotlib.pyplot as plt
```

```
In [12]: epochs = 25

         start = time()

         train_datagen_augmentation = ImageDataGenerator(
                         rotation_range=40,
                         width_shift_range=0.2,
                         height_shift_range=0.2,
                         shear_range=0.2,
                         zoom_range=0.2,
                         horizontal_flip = True)

         train_datagen_augmentation.fit(train_tensors)

         batch_size = 20

         checkpointer = ModelCheckpoint(filepath='saved_models/weights.best.from_scratch.hdf5'
                                        verbose=1, save_best_only=True)


         history = model.fit_generator(train_datagen_augmentation.flow(train_tensors, train_ta
                     steps_per_epoch=train_tensors.shape[0] // batch_size,
                     epochs=epochs,
                     verbose=1,
                     callbacks=[checkpointer],
                     validation_data=(valid_tensors, valid_targets)
                     )

         end = time()
         total_time = end - start
         print("The total computation time is {} ".format(total_time/60), " minutes")
```

```
# summarize history for accuracy
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```

```
WARNING:tensorflow:From /opt/anaconda3/lib/python3.7/site-packages/tensorflow/python/ops/math_
Instructions for updating:
Use tf.cast instead.
Epoch 1/25
334/334 [==============================] - 232s 695ms/step - loss: 4.8908 - acc: 0.0090 - val_

Epoch 00001: val_loss improved from inf to 4.88222, saving model to saved_models/weights.best.
Epoch 2/25
334/334 [==============================] - 232s 694ms/step - loss: 4.8662 - acc: 0.0145 - val_

Epoch 00002: val_loss improved from 4.88222 to 4.87644, saving model to saved_models/weights.b
Epoch 3/25
334/334 [==============================] - 230s 688ms/step - loss: 4.8313 - acc: 0.0193 - val_

Epoch 00003: val_loss improved from 4.87644 to 4.81856, saving model to saved_models/weights.b
Epoch 4/25
334/334 [==============================] - 229s 687ms/step - loss: 4.7926 - acc: 0.0204 - val_

Epoch 00004: val_loss improved from 4.81856 to 4.77730, saving model to saved_models/weights.b
Epoch 5/25
334/334 [==============================] - 230s 689ms/step - loss: 4.7530 - acc: 0.0247 - val_

Epoch 00005: val_loss did not improve from 4.77730
Epoch 6/25
334/334 [==============================] - 231s 692ms/step - loss: 4.6940 - acc: 0.0287 - val_

Epoch 00006: val_loss improved from 4.77730 to 4.74469, saving model to saved_models/weights.b
Epoch 7/25
334/334 [==============================] - 229s 685ms/step - loss: 4.6351 - acc: 0.0350 - val_

Epoch 00007: val_loss improved from 4.74469 to 4.63362, saving model to saved_models/weights.b
Epoch 8/25
334/334 [==============================] - 229s 685ms/step - loss: 4.5909 - acc: 0.0356 - val_l

Epoch 00008: val_loss improved from 4.63362 to 4.56705, saving model to saved_models/weights.be
Epoch 9/25
334/334 [==============================] - 229s 686ms/step - loss: 4.5084 - acc: 0.0401 - val_l

Epoch 00009: val_loss did not improve from 4.56705
Epoch 10/25
334/334 [==============================] - 229s 686ms/step - loss: 4.4378 - acc: 0.0493 - val_l

Epoch 00010: val_loss improved from 4.56705 to 4.42809, saving model to saved_models/weights.be
Epoch 11/25
334/334 [==============================] - 228s 684ms/step - loss: 4.3924 - acc: 0.0509 - val_l

Epoch 00011: val_loss did not improve from 4.42809
Epoch 12/25
334/334 [==============================] - 228s 684ms/step - loss: 4.3421 - acc: 0.0540 - val_l

Epoch 00012: val_loss improved from 4.42809 to 4.36327, saving model to saved_models/weights.be
Epoch 13/25
334/334 [==============================] - 229s 684ms/step - loss: 4.3239 - acc: 0.0576 - val_l

Epoch 00013: val_loss improved from 4.36327 to 4.34948, saving model to saved_models/weights.be
Epoch 14/25
334/334 [==============================] - 229s 685ms/step - loss: 4.3009 - acc: 0.0581 - val_l

Epoch 00014: val_loss did not improve from 4.34948
Epoch 15/25
334/334 [==============================] - 228s 682ms/step - loss: 4.2825 - acc: 0.0639 - val_l

Epoch 00015: val_loss did not improve from 4.34948
Epoch 16/25
334/334 [==============================] - 228s 683ms/step - loss: 4.2528 - acc: 0.0623 - val_l

Epoch 00016: val_loss improved from 4.34948 to 4.34671, saving model to saved_models/weights.be
Epoch 17/25
334/334 [==============================] - 228s 684ms/step - loss: 4.2382 - acc: 0.0675 - val_l

Epoch 00017: val_loss did not improve from 4.34671
Epoch 18/25
334/334 [==============================] - 229s 686ms/step - loss: 4.2153 - acc: 0.0663 - val_l

Epoch 00018: val_loss did not improve from 4.34671
Epoch 19/25
334/334 [==============================] - 228s 683ms/step - loss: 4.2063 - acc: 0.0705 - val_l

Epoch 00019: val_loss did not improve from 4.34671
Epoch 20/25
334/334 [==============================] - 230s 688ms/step - loss: 4.1958 - acc: 0.0681 - val_l

Epoch 00020: val_loss improved from 4.34671 to 4.30369, saving model to saved_models/weights.be
Epoch 21/25
334/334 [==============================] - 230s 688ms/step - loss: 4.1745 - acc: 0.0729 - val_l

Epoch 00021: val_loss improved from 4.30369 to 4.28222, saving model to saved_models/weights.be
Epoch 22/25
334/334 [==============================] - 230s 688ms/step - loss: 4.1563 - acc: 0.0762 - val_l

Epoch 00022: val_loss did not improve from 4.28222
Epoch 23/25
334/334 [==============================] - 229s 685ms/step - loss: 4.1333 - acc: 0.0771 - val_l

Epoch 00023: val_loss did not improve from 4.28222
Epoch 24/25
334/334 [==============================] - 231s 691ms/step - loss: 4.1287 - acc: 0.0814 - val_l

Epoch 00024: val_loss did not improve from 4.28222
Epoch 25/25
334/334 [==============================] - 230s 689ms/step - loss: 4.1187 - acc: 0.0763 - val_l

Epoch 00025: val_loss did not improve from 4.28222
The total computation time is 95.66487312316895   minutes
```





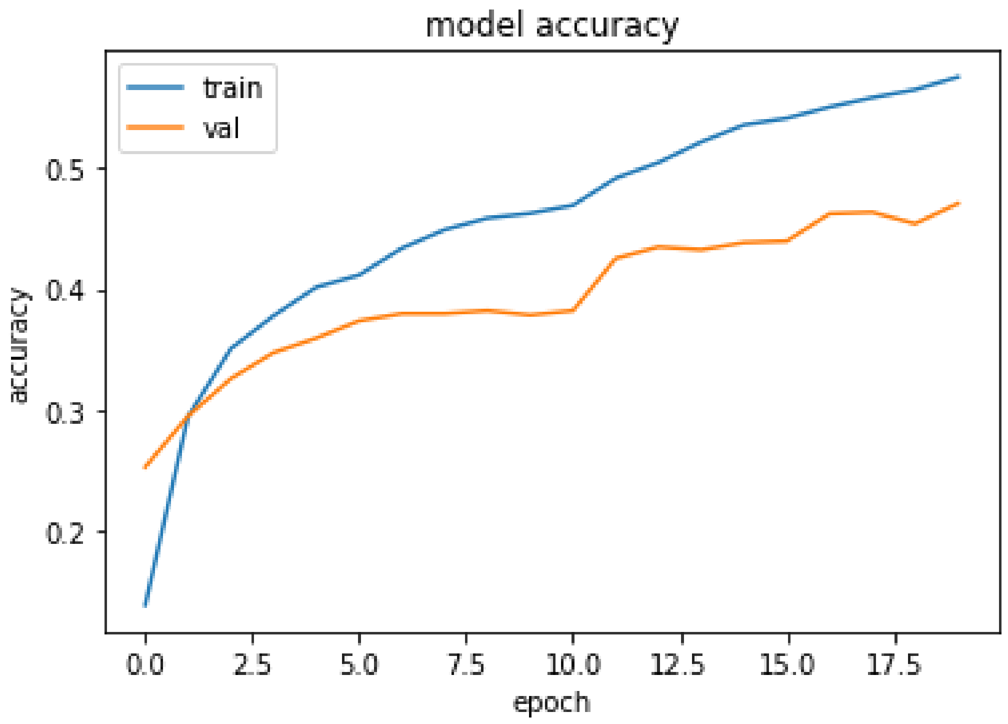
```
In [13]: model.load_weights('saved_models/weights.best.from_scratch.hdf5')

In [14]: dog_breed_predictions = [np.argmax(model.predict(np.expand_dims(tensor, axis=0))) for

         test_accuracy = 100*np.sum(np.array(dog_breed_predictions)==np.argmax(test_targets, a
         print('Test accuracy: %.4f%%' % test_accuracy)

Test accuracy: 7.0574%


In [15]: bottleneck_features = np.load('bottleneck_features/DogVGG16Data.npz')
         train_VGG16 = bottleneck_features['train']
         valid_VGG16 = bottleneck_features['valid']
         test_VGG16 = bottleneck_features['test']

In [16]: VGG16_model = Sequential()
         VGG16_model.add(GlobalAveragePooling2D(input_shape=train_VGG16.shape[1:]))
         VGG16_model.add(Dense(133, activation='softmax'))

         VGG16_model.summary()
```

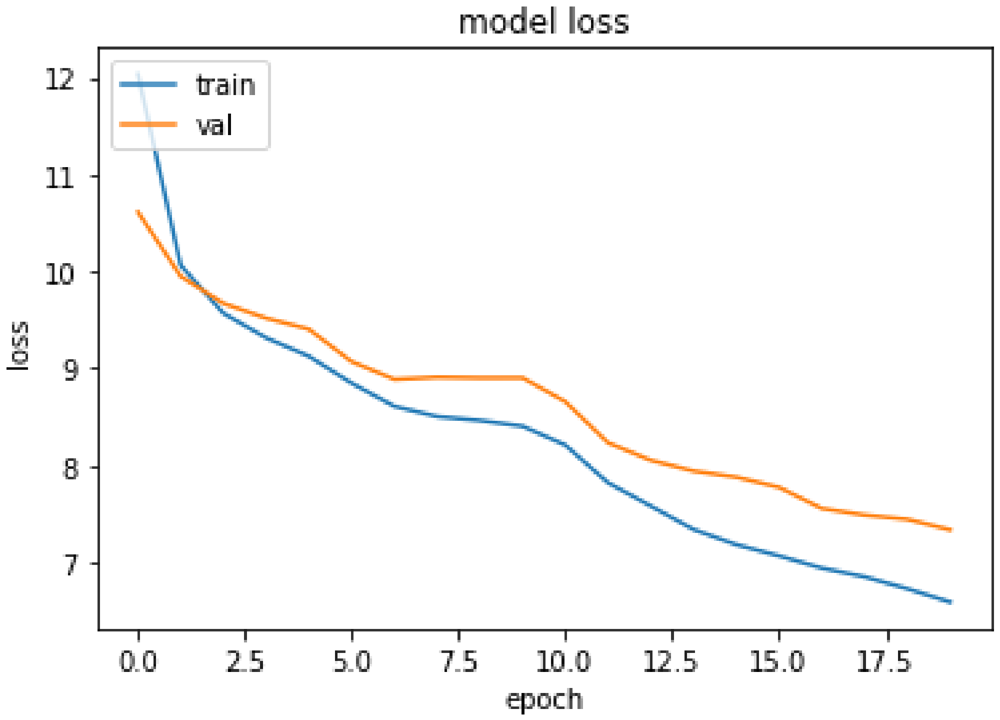
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
global_average_pooling2d_2 ( (None, 512)               0

_________________________________________________________________
dense_3 (Dense)              (None, 133)               68229
=================================================================
Total params: 68,229
Trainable params: 68,229
Non-trainable params: 0

_________________________________________________________________
```

```
In [17]: VGG16_model.compile(loss='categorical_crossentropy', optimizer='rmsprop', metrics=['a
```

```
In [18]: checkpointer = ModelCheckpoint(filepath='saved_models/weights.best.VGG16.hdf5',
                                        verbose=1, save_best_only=True)

         history = VGG16_model.fit(train_VGG16, train_targets,
                   validation_data=(valid_VGG16, valid_targets),
                   epochs=20, batch_size=20, callbacks=[checkpointer], verbose=1)

         # summarize history for accuracy
         plt.plot(history.history['acc'])
         plt.plot(history.history['val_acc'])
         plt.title('model accuracy')
         plt.ylabel('accuracy')
         plt.xlabel('epoch')
         plt.legend(['train', 'val'], loc='upper left')
         plt.show()
         # summarize history for loss
         plt.plot(history.history['loss'])
         plt.plot(history.history['val_loss'])
         plt.title('model loss')
         plt.ylabel('loss')
         plt.xlabel('epoch')
         plt.legend(['train', 'val'], loc='upper left')
         plt.show()
```

```
Train on 6680 samples, validate on 835 samples
Epoch 1/20
6680/6680 [==============================] - 3s 410us/step - loss: 12.0373 - acc: 0.1383 - val

Epoch 00001: val_loss improved from inf to 10.61376, saving model to saved_models/weights.best
Epoch 2/20
6680/6680 [==============================] - 2s 285us/step - loss: 10.0652 - acc: 0.2942 - val

Epoch 00002: val_loss improved from 10.61376 to 9.95138, saving model to saved_models/weights.
Epoch 3/20
6680/6680 [==============================] - 2s 260us/step - loss: 9.5697 - acc: 0.3506 - val_

Epoch 00003: val_loss improved from 9.95138 to 9.67160, saving model to saved_models/weights.b
Epoch 4/20
6680/6680 [==============================] - 2s 265us/step - loss: 9.3159 - acc: 0.3778 - val_

Epoch 00004: val_loss improved from 9.67160 to 9.51881, saving model to saved_models/weights.b
Epoch 5/20
6680/6680 [==============================] - 2s 277us/step - loss: 9.1257 - acc: 0.4015 - val_

Epoch 00005: val_loss improved from 9.51881 to 9.40568, saving model to saved_models/weights.b
Epoch 6/20
6680/6680 [==============================] - 2s 279us/step - loss: 8.8481 - acc: 0.4115 - val_

Epoch 00006: val_loss improved from 9.40568 to 9.07039, saving model to saved_models/weights.b
Epoch 7/20
6680/6680 [==============================] - 2s 266us/step - loss: 8.6080 - acc: 0.4337 - val_

Epoch 00007: val_loss improved from 9.07039 to 8.88713, saving model to saved_models/weights.b
Epoch 8/20
6680/6680 [==============================] - 2s 272us/step - loss: 8.5025 - acc: 0.4490 - val_

Epoch 00008: val_loss did not improve from 8.88713
Epoch 9/20
6680/6680 [==============================] - 2s 252us/step - loss: 8.4619 - acc: 0.4585 - val_

Epoch 00009: val_loss did not improve from 8.88713
Epoch 10/20
6680/6680 [==============================] - 2s 261us/step - loss: 8.4026 - acc: 0.4626 - val_

Epoch 00010: val_loss did not improve from 8.88713
Epoch 11/20
6680/6680 [==============================] - 2s 269us/step - loss: 8.2087 - acc: 0.4692 - val_

Epoch 00011: val_loss improved from 8.88713 to 8.65661, saving model to saved_models/weights.b
Epoch 12/20
6680/6680 [==============================] - 2s 241us/step - loss: 7.8210 - acc: 0.4916 - val_

Epoch 00012: val_loss improved from 8.65661 to 8.23185, saving model to saved_models/weights.b
Epoch 13/20
6680/6680 [==============================] - 2s 241us/step - loss: 7.5790 - acc: 0.5046 - val_

Epoch 00013: val_loss improved from 8.23185 to 8.05166, saving model to saved_models/weights.b
Epoch 14/20
6680/6680 [==============================] - 2s 248us/step - loss: 7.3372 - acc: 0.5217 - val_

Epoch 00014: val_loss improved from 8.05166 to 7.94102, saving model to saved_models/weights.b
Epoch 15/20
6680/6680 [==============================] - 2s 280us/step - loss: 7.1807 - acc: 0.5358 - val_l

Epoch 00015: val_loss improved from 7.94102 to 7.87815, saving model to saved_models/weights.be
Epoch 16/20
6680/6680 [==============================] - 2s 271us/step - loss: 7.0647 - acc: 0.5413 - val_l

Epoch 00016: val_loss improved from 7.87815 to 7.77322, saving model to saved_models/weights.be
Epoch 17/20
6680/6680 [==============================] - 2s 269us/step - loss: 6.9388 - acc: 0.5506 - val_l

Epoch 00017: val_loss improved from 7.77322 to 7.55449, saving model to saved_models/weights.be
Epoch 18/20
6680/6680 [==============================] - 2s 278us/step - loss: 6.8488 - acc: 0.5584 - val_l

Epoch 00018: val_loss improved from 7.55449 to 7.48652, saving model to saved_models/weights.be
Epoch 19/20
6680/6680 [==============================] - 2s 267us/step - loss: 6.7254 - acc: 0.5650 - val_l

Epoch 00019: val_loss improved from 7.48652 to 7.44242, saving model to saved_models/weights.be
Epoch 20/20
6680/6680 [==============================] - 2s 254us/step - loss: 6.5878 - acc: 0.5753 - val_l

Epoch 00020: val_loss improved from 7.44242 to 7.33436, saving model to saved_models/weights.be
```





```
In [19]: VGG16_model.load_weights('saved_models/weights.best.VGG16.hdf5')

In [20]: VGG16_predictions = [np.argmax(VGG16_model.predict(np.expand_dims(feature, axis=0))) 

         test_accuracy = 100*np.sum(np.array(VGG16_predictions)==np.argmax(test_targets, axis=
         print('Test accuracy: %.4f%%' % test_accuracy)

Test accuracy: 47.0096%


In [21]: import os
         import zipfile
         import tarfile
         import requests

         def download_file(url, path='./'):
             filename = url.split('/')[-1]
             print('Downloading {}'.format(filename))
             path = os.path.join(path, filename)
             r = requests.get(url, stream=True)
             with open(path, 'wb') as f:
                 for chunk in r.iter_content(chunk_size=1024):
```

```
                if chunk: # filter out keep-alive new chunks
                    f.write(chunk)
        print('Download complete')
        return filename

    def extract(archive, folder):
        print('Extracting {}'.format(archive))

        if (archive.endswith('tgz')):
            tar = tarfile.open(archive, 'r:gz')
            tar.extractall()
            tar.close()
        elif (archive.endswith('zip')):
            with zipfile.ZipFile(archive, 'r') as zip_ref:
                zip_ref.extractall()
        else:
            print('Archive type {} not recognized'.format(archive))

        if os.path.isdir(folder):
            print('Extracting complete'.format(archive))
        else:
            print('Extracting failed'.format(archive))

    def download_extract(url, folder, force_download=False):
        filename = url.split('/')[-1]
        downloadPath = os.path.join(os.getcwd(), folder)
        if os.path.isdir(downloadPath) is False:
            if os.path.exists(filename):
                if force_download is False:
                    print('File {} found skipping download'.format(filename))
                else:
                    print('Forcing download of {}'.format(filename))
                    download_file(url)
                extract(filename, downloadPath)
            else:
                download_file(url)
                extract(filename, downloadPath)
```

```
In [22]: """
         bottleneckFeaturesXceptionUrl = "https://s3-us-west-1.amazonaws.com/udacity-aind/dog-
         bottleneckFeaturesFolder = "bottleneck_features"
         download_file(bottleneckFeaturesXceptionUrl, bottleneckFeaturesFolder)
         """
```

```
Out[22]: '\nbottleneckFeaturesXceptionUrl = "https://s3-us-west-1.amazonaws.com/udacity-aind/do
```

```
In [23]: bottleneck_features = np.load('bottleneck_features/DogXceptionData.npz')
         train_Xception = bottleneck_features['train']
         valid_Xception = bottleneck_features['valid']
         test_Xception = bottleneck_features['test']
```

```
In [24]: from keras.layers import Dense, Flatten, GlobalAveragePooling2D, Dropout
         from keras.layers.advanced_activations import ELU
         from keras.layers.normalization import BatchNormalization

         Xception_model = Sequential()
         BatchNormalization(axis=-1)
         Xception_model.add(GlobalAveragePooling2D(input_shape=train_Xception.shape[1:] ))

         Xception_model.add(Dropout(0.4))

         Xception_model.add(Dense(64, activation="relu"))

         Xception_model.add(Dropout(0.3))

         Xception_model.add(Dense(133, activation="softmax"))

         Xception_model.summary()
```

```
WARNING:tensorflow:From /opt/anaconda3/lib/python3.7/site-packages/keras/backend/tensorflow_ba
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
global_average_pooling2d_3 ( (None, 2048)              0
_________________________________________________________________
dropout_1 (Dropout)          (None, 2048)              0
_________________________________________________________________
dense_4 (Dense)              (None, 64)                131136
_________________________________________________________________
dropout_2 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_5 (Dense)              (None, 133)               8645
=================================================================
Total params: 139,781
Trainable params: 139,781
Non-trainable params: 0
_________________________________________________________________
```

```
In [25]: Xception_model.compile(loss="categorical_crossentropy",
                      optimizer="rmsprop",
                       metrics=["accuracy"])
```

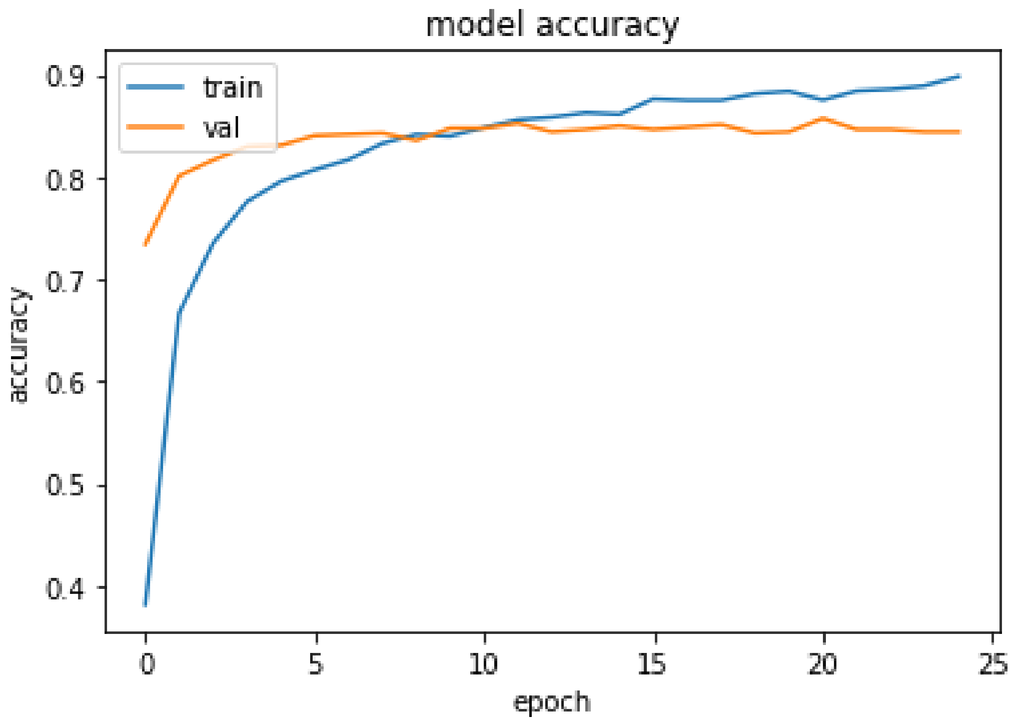
```
In [26]: from keras.callbacks import ModelCheckpoint
         from keras.preprocessing.image import ImageDataGenerator
```

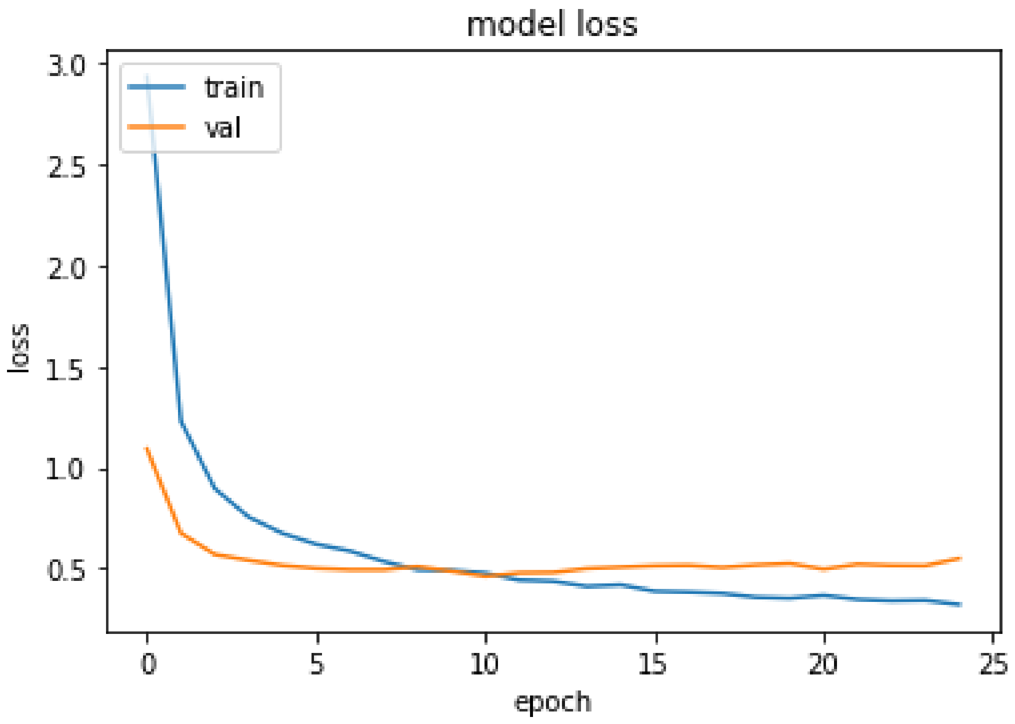
```
train_datagen_augmentation_2 = ImageDataGenerator(
                rotation_range=10,
                width_shift_range=0.2,
                height_shift_range=0.2,
                shear_range=0.2,
                zoom_range=0.1,
                horizontal_flip = True)

train_datagen_augmentation_2.fit(train_tensors)

epochs=25

batch_size=65

checkpointer = ModelCheckpoint(filepath='saved_models/weights.best.Xception.hdf5',
                               verbose=1, save_best_only=True)

history = Xception_model.fit(train_Xception, train_targets,
                validation_data=(valid_Xception, valid_targets),
                epochs=epochs,
                   callbacks=[checkpointer],
                   verbose=1
                  )

# summarize history for accuracy
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```

```
Train on 6680 samples, validate on 835 samples
Epoch 1/25
6680/6680 [==============================] - 4s 604us/step - loss: 2.9312 - acc: 0.3813 - val_

Epoch 00001: val_loss improved from inf to 1.09148, saving model to saved_models/weights.best.
Epoch 2/25
```

```
6680/6680 [==============================] - 3s 384us/step - loss: 1.2285 - acc: 0.6669 - val_l

Epoch 00002: val_loss improved from 1.09148 to 0.67553, saving model to saved_models/weights.be
Epoch 3/25
6680/6680 [==============================] - 3s 380us/step - loss: 0.8962 - acc: 0.7353 - val_l

Epoch 00003: val_loss improved from 0.67553 to 0.57020, saving model to saved_models/weights.be
Epoch 4/25
6680/6680 [==============================] - 3s 378us/step - loss: 0.7555 - acc: 0.7759 - val_l

Epoch 00004: val_loss improved from 0.57020 to 0.54136, saving model to saved_models/weights.be
Epoch 5/25
6680/6680 [==============================] - 3s 376us/step - loss: 0.6748 - acc: 0.7958 - val_l

Epoch 00005: val_loss improved from 0.54136 to 0.51559, saving model to saved_models/weights.be
Epoch 6/25
6680/6680 [==============================] - 3s 382us/step - loss: 0.6208 - acc: 0.8072 - val_l

Epoch 00006: val_loss improved from 0.51559 to 0.50151, saving model to saved_models/weights.be
Epoch 7/25
6680/6680 [==============================] - 3s 386us/step - loss: 0.5869 - acc: 0.8171 - val_l

Epoch 00007: val_loss improved from 0.50151 to 0.49476, saving model to saved_models/weights.be
Epoch 8/25
6680/6680 [==============================] - 3s 388us/step - loss: 0.5354 - acc: 0.8331 - val_l

Epoch 00008: val_loss did not improve from 0.49476
Epoch 9/25
6680/6680 [==============================] - 3s 376us/step - loss: 0.4941 - acc: 0.8418 - val_l

Epoch 00009: val_loss did not improve from 0.49476
Epoch 10/25
6680/6680 [==============================] - 3s 376us/step - loss: 0.4912 - acc: 0.8403 - val_l

Epoch 00010: val_loss improved from 0.49476 to 0.48749, saving model to saved_models/weights.be
Epoch 11/25
6680/6680 [==============================] - 2s 369us/step - loss: 0.4783 - acc: 0.8485 - val_l

Epoch 00011: val_loss improved from 0.48749 to 0.46186, saving model to saved_models/weights.be
Epoch 12/25
6680/6680 [==============================] - 3s 381us/step - loss: 0.4432 - acc: 0.8561 - val_l

Epoch 00012: val_loss did not improve from 0.46186
Epoch 13/25
6680/6680 [==============================] - 3s 385us/step - loss: 0.4377 - acc: 0.8587 - val_l

Epoch 00013: val_loss did not improve from 0.46186
Epoch 14/25
```

```
6680/6680 [==============================] - 3s 396us/step - loss: 0.4122 - acc: 0.8630 - val_

Epoch 00014: val_loss did not improve from 0.46186
Epoch 15/25
6680/6680 [==============================] - 3s 377us/step - loss: 0.4199 - acc: 0.8615 - val_

Epoch 00015: val_loss did not improve from 0.46186
Epoch 16/25
6680/6680 [==============================] - 3s 388us/step - loss: 0.3880 - acc: 0.8766 - val_

Epoch 00016: val_loss did not improve from 0.46186
Epoch 17/25
6680/6680 [==============================] - 3s 377us/step - loss: 0.3842 - acc: 0.8753 - val_

Epoch 00017: val_loss did not improve from 0.46186
Epoch 18/25
6680/6680 [==============================] - 4s 612us/step - loss: 0.3774 - acc: 0.8753 - val_

Epoch 00018: val_loss did not improve from 0.46186
Epoch 19/25
6680/6680 [==============================] - 3s 409us/step - loss: 0.3585 - acc: 0.8817 - val_

Epoch 00019: val_loss did not improve from 0.46186
Epoch 20/25
6680/6680 [==============================] - 3s 394us/step - loss: 0.3537 - acc: 0.8841 - val_

Epoch 00020: val_loss did not improve from 0.46186
Epoch 21/25
6680/6680 [==============================] - 3s 383us/step - loss: 0.3678 - acc: 0.8756 - val_

Epoch 00021: val_loss did not improve from 0.46186
Epoch 22/25
6680/6680 [==============================] - 3s 387us/step - loss: 0.3477 - acc: 0.8844 - val_

Epoch 00022: val_loss did not improve from 0.46186
Epoch 23/25
6680/6680 [==============================] - 3s 378us/step - loss: 0.3411 - acc: 0.8861 - val_

Epoch 00023: val_loss did not improve from 0.46186
Epoch 24/25
6680/6680 [==============================] - 3s 386us/step - loss: 0.3432 - acc: 0.8894 - val_

Epoch 00024: val_loss did not improve from 0.46186
Epoch 25/25
6680/6680 [==============================] - 3s 381us/step - loss: 0.3224 - acc: 0.8984 - val_

Epoch 00025: val_loss did not improve from 0.46186
```


model accuracy
accuracy
epoch
train
val
0.4
0.5
0.6
0.7
0.8
0.9
0
5
10
15
20
25


model loss
loss
epoch
train
val
0.5
1.0
1.5
2.0
2.5
3.0
0
5
10
15
20
25

```
In [27]: Xception_model.load_weights("saved_models/weights.best.Xception.hdf5")

In [28]: Xception_predictions = [np.argmax(Xception_model.predict(np.expand_dims(feature, axis

         test_accuracy = 100 * np.sum(np.array(Xception_predictions) == np.argmax(test_targets

         print("Xception Test accuracy: %.4f%%" % test_accuracy)

Xception Test accuracy: 83.3732%
```