Transfer Learning

Import Libraries

```
%matplotlib inline

import numpy as np
import pandas as pd
import keras
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Model
from keras.layers import Input, Flatten, Dense
from keras.callbacks import Callback, ModelCheckpoint
from keras.preprocessing import image
from keras.models import load_model

from keras.applications.vgg16 import VGG16
from keras.applications.vgg16 import preprocess_input

import warnings
warnings.filterwarnings('ignore')
```

Load the pretrained Network

```
model_vgg16_conv = VGG16(weights='imagenet', include_top=False)
```

Freeze the layers

```
for layer in model_vgg16_conv.layers:
    layer.trainable = False
```

Training parameters

```
img_width, img_height = 150, 150
train_data_dir = '/content/drive/MyDrive/train'
val_data_dir = '/content/drive/MyDrive/validation'
model_weights_file = 'vgg16-xfer-weights.h5'
nb_train_samples = 4
nb_val_samples = 4
nb_epochs = 5
```

Build a classification model on top of Base Network

```
input = Input(shape=(img_width, img_height, 3))
output_vgg16_conv = model_vgg16_conv(input)
x = Flatten()(output_vgg16_conv)
x = Dense(64, activation='relu')(x)
x = Dense(2, activation='softmax')(x)
model = Model(inputs=input, outputs=x)

model.summary()
```

```
Model: "model_6"
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_15 (InputLayer)        [(None, 150, 150, 3)]     0
_________________________________________________________________
vgg16 (Functional)           (None, None, None, 512)   14714688
_________________________________________________________________
flatten_8 (Flatten)          (None, 8192)              0
_________________________________________________________________
dense_16 (Dense)             (None, 64)                524352
_________________________________________________________________
dense_17 (Dense)             (None, 2)                 130
=================================================================
Total params: 15,239,170
Trainable params: 524,482
Non-trainable params: 14,714,688
_________________________________________________________________
```

```
model.compile(loss='categorical_crossentropy', optimizer='rmsprop', metrics=['accuracy'])
```

Dataset Preparation

```
train_datagen = ImageDataGenerator(rescale=1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(train_data_dir, target_size=(img_width, img_height),
                                                    batch_size=1, class_mode='categorical')
validation_generator = test_datagen.flow_from_directory(val_data_dir, target_size=(img_width, img_height),
                                                        batch_size=1,class_mode='categorical')
```

```
Found 4 images belonging to 2 classes.
Found 4 images belonging to 2 classes.
```

Training

```
#callbacks = [ModelCheckpoint(model_weights_file, monitor='val_acc', save_best_only=True)]
#callbacks = keras.callbacks.ModelCheckpoint("vgg16-xfer-weights.h5", monitor='val_acc', save_best_only=True)
callbacks = keras.callbacks.ModelCheckpoint("vgg16-xfer-weights.h5",save_best_only=True)
history = model.fit_generator( train_generator, callbacks = callbacks, steps_per_epoch=nb_train_samples,epochs=nb_epochs, va

print('Training Completed!')
```

```
Epoch 1/5
4/4 [==============================] - 2s 497ms/step - loss: 3.6369 - accuracy: 0.2667 - val_loss: 0.8315 - val_accura
Epoch 2/5
4/4 [==============================] - 2s 437ms/step - loss: 0.2915 - accuracy: 1.0000 - val_loss: 1.1149 - val_accura
Epoch 3/5
4/4 [==============================] - 2s 467ms/step - loss: 0.2888 - accuracy: 1.0000 - val_loss: 0.5462 - val_accura
Epoch 4/5
4/4 [==============================] - 2s 436ms/step - loss: 0.0448 - accuracy: 1.0000 - val_loss: 0.6532 - val_accura
Epoch 5/5
4/4 [==============================] - 1s 419ms/step - loss: 0.0697 - accuracy: 1.0000 - val_loss: 0.6466 - val_accura
Training Completed!
```

```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

```
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```

Test the model

```
img_path = '/content/drive/MyDrive/dog.jpg'
label = ['Cat','Dog']
img = image.load_img(img_path, target_size=(150, 150))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
x = preprocess_input(x)

features = model.predict(x)
thresholded = (features>0.5)*1
ind = np.argmax(thresholded)
print('Predicted Array:',thresholded)
print('Predicted Label:',label[ind])
```

```
WARNING:tensorflow:7 out of the last 11 calls to <function Model.make_predict_function.<locals>.predict_function at 0x
Predicted Array: [[0 1]]
Predicted Label: Dog
```

Saved Model Deployment

```
model = load_model('/content/drive/MyDrive/vgg16-xfer-weights.h5')
model.summary()
```

```
Model: "model_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_13 (InputLayer)        [(None, 150, 150, 3)]     0
_________________________________________________________________
vgg16 (Functional)           (None, None, None, 512)   14714688
```

```
_________________________________________________________________
flatten_7 (Flatten)          (None, 8192)              0         
_________________________________________________________________
dense_14 (Dense)             (None, 64)                524352    
_________________________________________________________________
dense_15 (Dense)             (None, 2)                 130       
=================================================================
Total params: 15,239,170
Trainable params: 524,482
Non-trainable params: 14,714,688
_________________________________________________________________
```

```
img_path = '/content/drive/MyDrive/cat.jpg'
label = ['Cat','Dog']
img = image.load_img(img_path, target_size=(150, 150))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
x = preprocess_input(x)

features = model.predict(x)
thresholded = (features>0.5)*1
ind = np.argmax(thresholded)
print('Predicted Array:',thresholded)
print('Predicted Label:',label[ind])
```

```
WARNING:tensorflow:7 out of the last 11 calls to <function Model.make_predict_function.<locals>.predict_function at 0x
Predicted Array: [[1 0]]
Predicted Label: Cat
```

Models for image classification with weights trained on ImageNet

Xception

VGG16

VGG19

ResNet50

InceptionV3

InceptionResNetV2

MobileNet

DenseNet

***NASNet ***
