In [1]:

```
#########################
#   INITIALIZATION
#########################

PATH = 'dataset/'
sz_shape = (224, 224)
batch_size = 64

import numpy as np
from keras.preprocessing.image import ImageDataGenerator
from keras.layers import Dense, GlobalAveragePooling2D
from keras.applications.resnet50 import ResNet50
from keras.applications.resnet50 import preprocess_input
from keras.models import Model
```

```
Using TensorFlow backend.
```

In [2]:

```
########################
#1   DATASET
########################

train_data_dir = f"{PATH}training_set"
validation_data_dir = f"{PATH}test_set"
```

In [3]:

```
#In the flow_from_directory method, the normalization is configured to apply to a batch of inputs,
#and you cannot manipulate a numpy array in that method. You will have to manually
#standardize each input x in the API provided.
#You can just inherit from the ImageDataGenerator class
#and override the function standardize to fit your data properly

class FixedImageDataGenerator(ImageDataGenerator):
    def standardize(self, x):
        if self.featurewise_center:
            x = ((x/255.) - 0.5) * 2.
        return x

train_datagen = FixedImageDataGenerator(preprocess_input, shear_range= 0.2,
                                        zoom_range=0.2,
                                        horizontal_flip=True)
test_datagen = FixedImageDataGenerator(preprocess_input)

train_generator = train_datagen.flow_from_directory(train_data_dir,
                                                    target_size=sz_shape,
                                                    batch_size=batch_size,
                                                    class_mode='binary')
validation_generator = test_datagen.flow_from_directory(validation_data_dir,
                                                        shuffle=False,
                                                        target_size=sz_shape,
                                                        batch_size=batch_size,
                                                        class_mode='binary')
```

```
Found 8000 images belonging to 2 classes.
Found 2000 images belonging to 2 classes.
```

In [4]:

```
########################
#2  MODEL
########################

base_model = ResNet50(weights='imagenet', include_top=False)
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation = 'relu')(x)
predictions = Dense(1, activation = 'sigmoid')(x)

model = Model(inputs=base_model.input, outputs=predictions)

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

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

In [5]:

```
########################
#3  TRAIN THE MODEL
########################


model.fit_generator(train_generator, train_generator.n // batch_size,
                    epochs = 1, validation_data = validation_generator,
                    validation_steps = validation_generator.n // batch_size)
```

```
Epoch 1/1
125/125 [==============================] - 604s 5s/step - loss: 0.3687 - acc: 0.9105 - val_loss: 1.5924
- val_acc: 0.5040
```

Out[5]:

```
<keras.callbacks.History at 0x1b617715358>
```

In [6]:

```
########################
#4  EVALUATE THE MODEL
########################

score = model.evaluate_generator(train_generator)

print('test loss: {:.4f}'.format(score[0]))
print('test acc: {:.4f}'.format(score[1]))
```