In [1]:
```
from keras.layers import Input, Lambda, Dense, Flatten
from keras.models import Model
from keras.applications.vgg19 import VGG19
from keras.applications.vgg19 import preprocess_input
from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
import numpy as np
from glob import glob
import matplotlib.pyplot as plt
```

In [2]:
```
IMAGE_SIZE = [224, 224]

train_path = 'E:\\chest_xray\\train\\'
valid_path = 'E:\\chest_xray\\test\\'
```

In [3]:
```
vgg = VGG19(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)
```

In [4]:
```
for layer in vgg.layers:
    layer.trainable = False
```

In [5]:
```
folders = glob('E:\\chest_xray\\train\\*')
```

In [6]:
```
x = Flatten()(vgg.output)
```

In [7]:
```
prediction = Dense(len(folders), activation='softmax')(x)
model = Model(inputs=vgg.input, outputs=prediction)
```

In [8]:
```
model.summary()
```

```
Model: "functional_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
```

```
block3_conv4 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv4 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv4 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
dense (Dense)                (None, 2)                 50178
=================================================================
Total params: 20,074,562
Trainable params: 50,178
Non-trainable params: 20,024,384
_________________________________________________________________
```

In [9]:

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

In [10]:

```
from keras.preprocessing.image import ImageDataGenerator

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

test_datagen = ImageDataGenerator(rescale = 1./255)
```

In [11]:

```
training_set = train_datagen.flow_from_directory('E:\\chest_xray\\train',
                                                 target_size = (224, 224),
                                                 batch_size = 32,
                                                 class_mode = 'categorical')
```

```
Found 5216 images belonging to 2 classes.
```

In [12]:

```
test_set = test_datagen.flow_from_directory('E:\\chest_xray\\test',
                                            target_size = (224, 224),
                                            batch_size = 32,
                                            class_mode = 'categorical')
```

```
Found 624 images belonging to 2 classes.
```

In [13]:

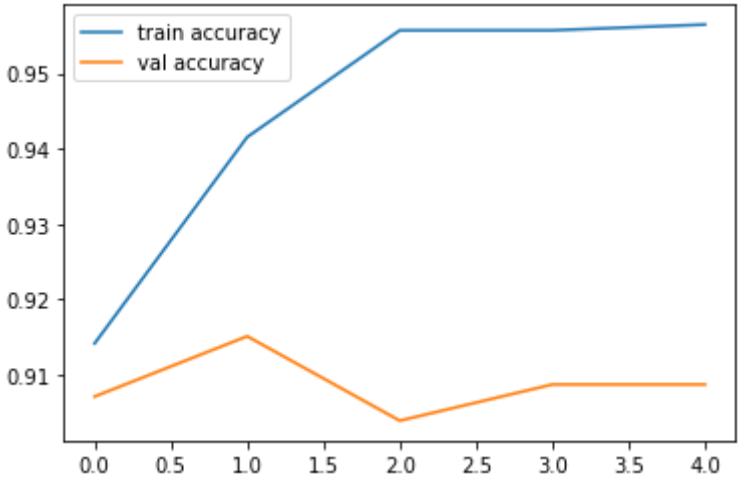
```
r = model.fit_generator(
  training_set,
  validation_data=test_set,
  epochs=5,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_set)
)
```

```
WARNING:tensorflow:From C:\Users\MCHOME\AppData\Local\Temp/ipykernel_1612/675562961.
py:6: Model.fit_generator (from tensorflow.python.keras.engine.training) is deprecat
ed and will be removed in a future version.
Instructions for updating:
Please use Model.fit, which supports generators.
Epoch 1/5
163/163 [==============================] - 5316s 33s/step - loss: 0.2128 - accuracy:
0.9141 - val_loss: 0.2569 - val_accuracy: 0.9071
Epoch 2/5
163/163 [==============================] - 6676s 41s/step - loss: 0.1604 - accuracy:
0.9415 - val_loss: 0.2923 - val_accuracy: 0.9151
Epoch 3/5
163/163 [==============================] - 7472s 46s/step - loss: 0.1191 - accuracy:
0.9557 - val_loss: 0.3394 - val_accuracy: 0.9038
Epoch 4/5
163/163 [==============================] - 7200s 44s/step - loss: 0.1250 - accuracy:
0.9557 - val_loss: 0.2921 - val_accuracy: 0.9087
Epoch 5/5
163/163 [==============================] - 7372s 45s/step - loss: 0.1122 - accuracy:
0.9565 - val_loss: 0.3187 - val_accuracy: 0.9087
```

In [14]:

```
plt.plot(r.history['loss'], label='train loss')
plt.plot(r.history['val_loss'], label='val loss')
plt.legend()
plt.show()
plt.savefig('LossVal_loss')

# plot the accuracy
plt.plot(r.history['accuracy'], label='train accuracy')
plt.plot(r.history['val_accuracy'], label='val accuracy')
plt.legend()
plt.show()
plt.savefig('AccVal_accuracy')
```




```
<Figure size 432x288 with 0 Axes>
```

In [15]:
```
import tensorflow as tf

from keras.models import load_model

model.save('E:\\chest_xray\\val\\model_vgg19.h5')
```

In [ ]:

In [ ]:

In [ ]: