```
import os
import numpy as np
import pandas as pd
import random
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

# Deep learning libraries
import keras.backend as K
from keras.models import Model, Sequential
from keras.layers import Input, Dense, Flatten, Dropout, BatchNormalization
from keras.layers import Conv2D, SeparableConv2D, MaxPool2D, LeakyReLU, Activation
from keras.optimizers import Adam
from keras.preprocessing.image import ImageDataGenerator
from google.colab import files
from keras.callbacks import ModelCheckpoint, ReduceLROnPlateau, EarlyStopping
import tensorflow as tf
print(tf.__version__)

# Setting seeds for reproducibility
seed = 232
np.random.seed(seed)
tf.random.set_seed(seed)
```

```
2.12.0
```

```
files.upload()
```

```
Choose Files kaggle.json
- kaggle.json(application/json) - 75 bytes, last modified: 5/17/2023 - 100% done
Saving kaggle.json to kaggle.json
```

```
import os
os.environ["KAGGLE_CONFIG_DIR"] = "/content"
```

```
!kaggle datasets download -d paultimothymooney/chest-xray-pneumonia
```

```
Warning: Your Kaggle API key is readable by other users on this system! To fix this, you can run 'chmod 600 /content/kaggle.json'
Downloading chest-xray-pneumonia.zip to /content
 99% 2.28G/2.29G [00:30<00:00, 53.7MB/s]
100% 2.29G/2.29G [00:30<00:00, 80.9MB/s]
```

```
!unzip \*.zip
```

```
  inflating: chest_xray/train/PNEUMONIA/person716_virus_1314.jpeg
  inflating: chest_xray/train/PNEUMONIA/person717_bacteria_2618.jpeg
  inflating: chest_xray/train/PNEUMONIA/person718_bacteria_2620.jpeg
  inflating: chest_xray/train/PNEUMONIA/person718_virus_1316.jpeg
  inflating: chest_xray/train/PNEUMONIA/person719_bacteria_2621.jpeg
  inflating: chest_xray/train/PNEUMONIA/person719_virus_1338.jpeg
  inflating: chest_xray/train/PNEUMONIA/person71_bacteria_347.jpeg
  inflating: chest_xray/train/PNEUMONIA/person71_bacteria_348.jpeg
  inflating: chest_xray/train/PNEUMONIA/person71_bacteria_349.jpeg
  inflating: chest_xray/train/PNEUMONIA/person71_bacteria_350.jpeg
  inflating: chest_xray/train/PNEUMONIA/person71_bacteria_351.jpeg
  inflating: chest_xray/train/PNEUMONIA/person720_bacteria_2622.jpeg
  inflating: chest_xray/train/PNEUMONIA/person720_virus_1339.jpeg
  inflating: chest_xray/train/PNEUMONIA/person721_bacteria_2623.jpeg
  inflating: chest_xray/train/PNEUMONIA/person721_virus_1340.jpeg
  inflating: chest_xray/train/PNEUMONIA/person722_virus_1341.jpeg
  inflating: chest_xray/train/PNEUMONIA/person723_bacteria_2625.jpeg
  inflating: chest_xray/train/PNEUMONIA/person723_virus_1342.jpeg
  inflating: chest_xray/train/PNEUMONIA/person724_bacteria_2626.jpeg
  inflating: chest_xray/train/PNEUMONIA/person724_virus_1343.jpeg
  inflating: chest_xray/train/PNEUMONIA/person724_virus_1344.jpeg
  inflating: chest_xray/train/PNEUMONIA/person725_bacteria_2627.jpeg
  inflating: chest_xray/train/PNEUMONIA/person726_bacteria_2628.jpeg
  inflating: chest_xray/train/PNEUMONIA/person727_bacteria_2629.jpeg
```

```
for dirpath,dirnames,filenames in os.walk("/content/chest_xray"):
    print(f"there are {len(dirnames)} directories and {len(filenames)} images in '{dirpath}'.")
```

```
 there are 5 directories and 0 images in '/content/chest_xray'.
 there are 2 directories and 0 images in '/content/chest_xray/val'.
 there are 0 directories and 8 images in '/content/chest_xray/val/PNEUMONIA'.
 there are 0 directories and 8 images in '/content/chest_xray/val/NORMAL'.
 there are 1 directories and 1 images in '/content/chest_xray/__MACOSX'.
 there are 3 directories and 3 images in '/content/chest_xray/__MACOSX/chest_xray'.
 there are 2 directories and 1 images in '/content/chest_xray/__MACOSX/chest_xray/val'.
 there are 0 directories and 9 images in '/content/chest_xray/__MACOSX/chest_xray/val/PNEUMONIA'.
 there are 0 directories and 9 images in '/content/chest_xray/__MACOSX/chest_xray/val/NORMAL'.
 there are 2 directories and 3 images in '/content/chest_xray/__MACOSX/chest_xray/train'.
 there are 0 directories and 3876 images in '/content/chest_xray/__MACOSX/chest_xray/train/PNEUMONIA'.
 there are 0 directories and 1342 images in '/content/chest_xray/__MACOSX/chest_xray/train/NORMAL'.
 there are 2 directories and 3 images in '/content/chest_xray/__MACOSX/chest_xray/test'.
 there are 0 directories and 390 images in '/content/chest_xray/__MACOSX/chest_xray/test/PNEUMONIA'.
 there are 0 directories and 234 images in '/content/chest_xray/__MACOSX/chest_xray/test/NORMAL'.
 there are 3 directories and 1 images in '/content/chest_xray/chest_xray'.
 there are 2 directories and 1 images in '/content/chest_xray/chest_xray/val'.
 there are 0 directories and 9 images in '/content/chest_xray/chest_xray/val/PNEUMONIA'.
 there are 0 directories and 9 images in '/content/chest_xray/chest_xray/val/NORMAL'.
 there are 2 directories and 1 images in '/content/chest_xray/chest_xray/train'.
 there are 0 directories and 3876 images in '/content/chest_xray/chest_xray/train/PNEUMONIA'.
 there are 0 directories and 1342 images in '/content/chest_xray/chest_xray/train/NORMAL'.
 there are 2 directories and 1 images in '/content/chest_xray/chest_xray/test'.
 there are 0 directories and 390 images in '/content/chest_xray/chest_xray/test/PNEUMONIA'.
 there are 0 directories and 234 images in '/content/chest_xray/chest_xray/test/NORMAL'.
 there are 2 directories and 0 images in '/content/chest_xray/train'.
 there are 0 directories and 3875 images in '/content/chest_xray/train/PNEUMONIA'.
 there are 0 directories and 1341 images in '/content/chest_xray/train/NORMAL'.
 there are 2 directories and 0 images in '/content/chest_xray/test'.
 there are 0 directories and 390 images in '/content/chest_xray/test/PNEUMONIA'.
 there are 0 directories and 234 images in '/content/chest_xray/test/NORMAL'.
```

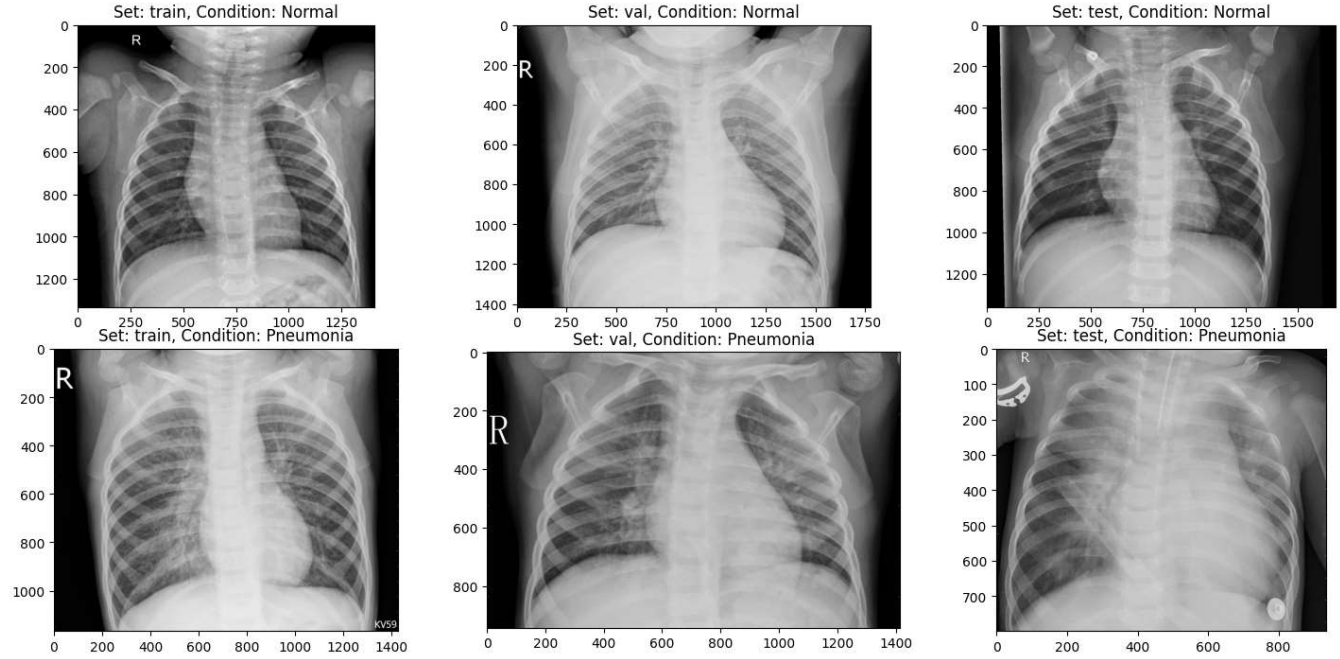
```
import pathlib
data_dir = pathlib.Path("/content/chest_xray")
class_names = np.array(sorted([item.name for item in data_dir.glob("*")]))
class_names
```

```
 array(['__MACOSX', 'chest_xray', 'test', 'train', 'val'], dtype='<U10')
```

```
import os
import matplotlib.pyplot as plt

input_path = '/content/chest_xray/chest_xray/'
print(input_path)

fig, ax = plt.subplots(2, 3, figsize=(15, 7))
ax = ax.ravel()
plt.tight_layout()

for i, _set in enumerate(['train', 'val', 'test']):
    set_path = os.path.join(input_path, _set)
    normal_path = os.path.join(set_path, 'NORMAL')
    pneumonia_path = os.path.join(set_path, 'PNEUMONIA')

    ax[i].imshow(plt.imread(os.path.join(normal_path, os.listdir(normal_path)[0])), cmap='gray')
    ax[i].set_title('Set: {}, Condition: Normal'.format(_set))
```

```
    ax[i+3].imshow(plt.imread(os.path.join(pneumonia_path, os.listdir(pneumonia_path)[0])), cmap='gray')
    ax[i+3].set_title('Set: {}, Condition: Pneumonia'.format(_set))

plt.show()
```

```
/content/chest_xray/chest_xray/
```




```
for _set in ['train', 'val', 'test']:
    n_normal = len(os.listdir(input_path + _set + '/NORMAL'))
    n_infect = len(os.listdir(input_path + _set + '/PNEUMONIA'))
    print('Set: {}, normal images: {}, pneumonia images: {}'.format(_set, n_normal, n_infect))
```

```
Set: train, normal images: 1342, pneumonia images: 3876
Set: val, normal images: 9, pneumonia images: 9
Set: test, normal images: 234, pneumonia images: 390
```

```
input_path = '/content/chest_xray/chest_xray/'

def process_data(img_dims, batch_size):
    # Data generation objects
    train_datagen = ImageDataGenerator(rescale=1./255, zoom_range=0.3, vertical_flip=True)
    test_val_datagen = ImageDataGenerator(rescale=1./255)

    # This is fed to the network in the specified batch sizes and image dimensions
    train_gen = train_datagen.flow_from_directory(
    directory=input_path+'train',
    target_size=(img_dims, img_dims),
    batch_size=batch_size,
    class_mode='binary',
    shuffle=True)

    test_gen = test_val_datagen.flow_from_directory(
    directory=input_path+'test',
    target_size=(img_dims, img_dims),
    batch_size=batch_size,
    class_mode='binary',
    shuffle=True)


    test_data = []
    test_labels = []

    for cond in ['/NORMAL/', '/PNEUMONIA/']:
        for img in (os.listdir(input_path + 'test' + cond)):
            img = plt.imread(input_path+'test'+cond+img)
            img = cv2.resize(img, (img_dims, img_dims))
            img = np.dstack([img, img, img])
            img = img.astype('float32') / 255
            if cond=='/NORMAL/':
                label = 0
            elif cond=='/PNEUMONIA/':
                label = 1
```

```
            test_data.append(img)
            test_labels.append(label)

    test_data = np.array(test_data)
    test_labels = np.array(test_labels)

    return train_gen, test_gen, test_data, test_labels


img_dims = 150
epochs = 5
batch_size = 32

# Getting the data
train_gen, test_gen, test_data, test_labels = process_data(img_dims, batch_size)
```

```
 Found 5216 images belonging to 2 classes.
 Found 624 images belonging to 2 classes.
```

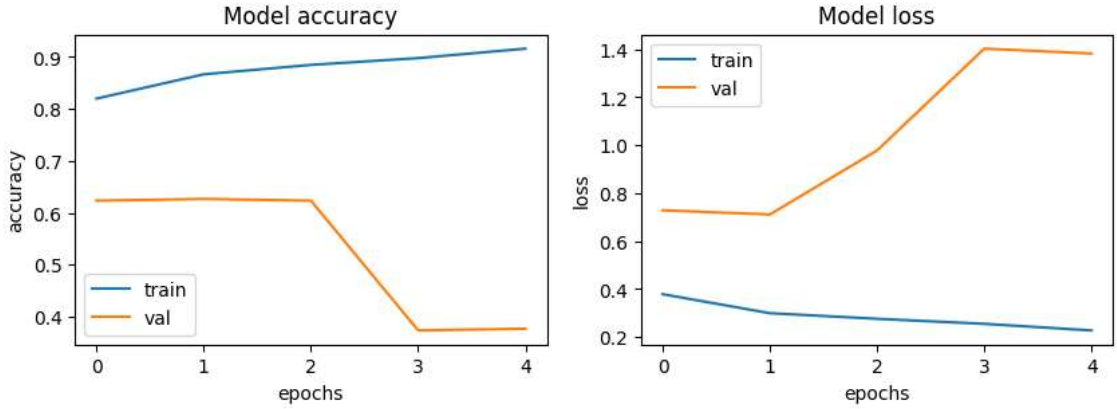
```
# Input layer
inputs = Input(shape=(img_dims, img_dims, 3))

# First conv block
x = Conv2D(filters=16, kernel_size=(3, 3), activation='relu', padding='same')(inputs)
x = Conv2D(filters=16, kernel_size=(3, 3), activation='relu', padding='same')(x)
x = MaxPool2D(pool_size=(2, 2))(x)

# Second conv block
x = SeparableConv2D(filters=32, kernel_size=(3, 3), activation='relu', padding='same')(x)
x = SeparableConv2D(filters=32, kernel_size=(3, 3), activation='relu', padding='same')(x)
x = BatchNormalization()(x)
x = MaxPool2D(pool_size=(2, 2))(x)

# Third conv block
x = SeparableConv2D(filters=64, kernel_size=(3, 3), activation='relu', padding='same')(x)
x = SeparableConv2D(filters=64, kernel_size=(3, 3), activation='relu', padding='same')(x)
x = BatchNormalization()(x)
x = MaxPool2D(pool_size=(2, 2))(x)

# Fourth conv block
x = SeparableConv2D(filters=128, kernel_size=(3, 3), activation='relu', padding='same')(x)
x = SeparableConv2D(filters=128, kernel_size=(3, 3), activation='relu', padding='same')(x)
x = BatchNormalization()(x)
x = MaxPool2D(pool_size=(2, 2))(x)
x = Dropout(rate=0.2)(x)

# Fifth conv block
x = SeparableConv2D(filters=256, kernel_size=(3, 3), activation='relu', padding='same')(x)
x = SeparableConv2D(filters=256, kernel_size=(3, 3), activation='relu', padding='same')(x)
x = BatchNormalization()(x)
x = MaxPool2D(pool_size=(2, 2))(x)
x = Dropout(rate=0.2)(x)

# FC layer
x = Flatten()(x)
x = Dense(units=512, activation='relu')(x)
x = Dropout(rate=0.7)(x)
x = Dense(units=128, activation='relu')(x)
x = Dropout(rate=0.5)(x)
x = Dense(units=64, activation='relu')(x)
x = Dropout(rate=0.3)(x)

# Output layer
output = Dense(units=1, activation='sigmoid')(x)

# Creating model and compiling
model = Model(inputs=inputs, outputs=output)
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Callbacks
checkpoint = ModelCheckpoint(filepath='best_weights.hdf5', save_best_only=True, save_weights_only=True)
lr_reduce = ReduceLROnPlateau(monitor='val_loss', factor=0.3, patience=2, verbose=2, mode='max')
early_stop = EarlyStopping(monitor='val_loss', min_delta=0.1, patience=1, mode='min')
```

```
# Fitting the model
hist = model.fit_generator(
           train_gen, steps_per_epoch=train_gen.samples // batch_size,
```

```
        epochs=epochs, validation_data=test_gen,
        validation_steps=test_gen.samples // batch_size, callbacks=[checkpoint, lr_reduce])
```

```
<ipython-input-13-4812e868c216>:2: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please
  hist = model.fit_generator(
Epoch 1/5
163/163 [==============================] - 392s 2s/step - loss: 0.3787 - accuracy: 0.8200 - val_loss: 0.7286 - val_accuracy: 0.6234
Epoch 2/5
163/163 [==============================] - 379s 2s/step - loss: 0.2995 - accuracy: 0.8668 - val_loss: 0.7110 - val_accuracy: 0.6266
Epoch 3/5
163/163 [==============================] - 371s 2s/step - loss: 0.2760 - accuracy: 0.8850 - val_loss: 0.9782 - val_accuracy: 0.6234
Epoch 4/5
163/163 [==============================] - 380s 2s/step - loss: 0.2551 - accuracy: 0.8980 - val_loss: 1.4029 - val_accuracy: 0.3734
Epoch 5/5
163/163 [==============================] - 374s 2s/step - loss: 0.2275 - accuracy: 0.9162 - val_loss: 1.3826 - val_accuracy: 0.3766
```

```
fig, ax = plt.subplots(1, 2, figsize=(10, 3))
ax = ax.ravel()

for i, met in enumerate(['accuracy', 'loss']):
    ax[i].plot(hist.history[met])
    ax[i].plot(hist.history['val_' + met])
    ax[i].set_title('Model {}'.format(met))
    ax[i].set_xlabel('epochs')
    ax[i].set_ylabel(met)
    ax[i].legend(['train', 'val'])
```




```
from tensorflow.keras.preprocessing import image
img = image.load_img(r"/content/tocheck2.jpeg", target_size=(150, 150))
img_array = image.img_to_array(img)
img_array = img_array / 255.0
img_batch = np.expand_dims(img_array, axis=0)


preds = model.predict(img_batch)
print(preds)

if preds[0][0] >0.5:
    print(" The Person is affected by Pneumonia")
else:
    print(" The Person is in Normal Condition")
```

```
1/1 [==============================] - 0s 177ms/step
[[0.7711995]]
 The Person is affected by Pneumonia
```

```
!pip install gradio
```

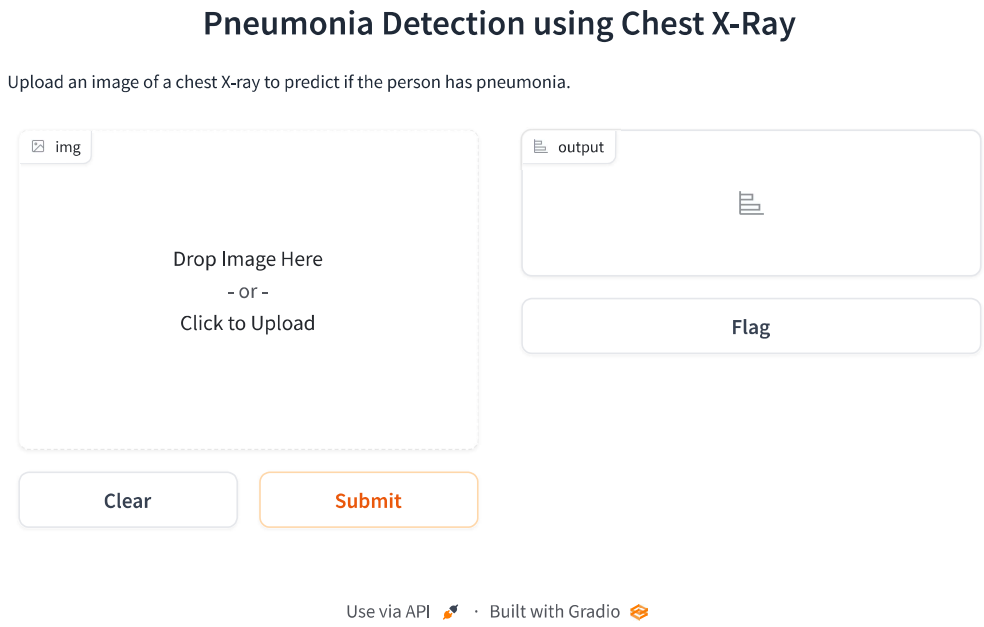
```
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.14.0->gradio) (3.12.0
Requirement already satisfied: tqdm>=4.42.1 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.14.0->gradio) (4
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-packages (from markdown-it-py[linkify]>=2.0.0->gradio
Collecting linkify-it-py<3,>=1 (from markdown-it-py[linkify]>=2.0.0->gradio)
  Downloading linkify_it_py-2.0.2-py3-none-any.whl (19 kB)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from pandas->gradio) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->gradio) (2022.7.1)
Requirement already satisfied: click>=7.0 in /usr/local/lib/python3.10/dist-packages (from uvicorn>=0.14.0->gradio) (8.1.3)
Collecting h11>=0.8 (from uvicorn>=0.14.0->gradio)
  Downloading h11-0.14.0-py3-none-any.whl (58 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.3/58.3 kB 7.8 MB/s eta 0:00:00
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->gradio) (23.1.0)
Requirement already satisfied: charset-normalizer<4.0,>=2.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp->gradio) (2
Collecting multidict<7.0,>=4.5 (from aiohttp->gradio)
  Downloading multidict-6.0.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (114 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 114.5/114.5 kB 14.9 MB/s eta 0:00:00
Collecting async-timeout<5.0,>=4.0.0a3 (from aiohttp->gradio)
  Downloading async_timeout-4.0.2-py3-none-any.whl (5.8 kB)
Collecting yarl<2.0,>=1.0 (from aiohttp->gradio)
  Downloading yarl-1.9.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (268 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 268.8/268.8 kB 29.3 MB/s eta 0:00:00
Collecting frozenlist>=1.1.1 (from aiohttp->gradio)
  Downloading frozenlist-1.3.3-cp310-cp310-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 149.6/149.6 kB 18.5 MB/s eta 0:00:00
Collecting aiosignal>=1.1.2 (from aiohttp->gradio)
  Downloading aiosignal-1.3.1-py3-none-any.whl (7.6 kB)
Collecting starlette<0.28.0,>=0.27.0 (from fastapi->gradio)
  Downloading starlette-0.27.0-py3-none-any.whl (66 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 67.0/67.0 kB 9.0 MB/s eta 0:00:00
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from httpx->gradio) (2022.12.7)
Collecting httpcore<0.18.0,>=0.15.0 (from httpx->gradio)
  Downloading httpcore-0.17.2-py3-none-any.whl (72 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 72.5/72.5 kB 7.7 MB/s eta 0:00:00
Requirement already satisfied: idna in /usr/local/lib/python3.10/dist-packages (from httpx->gradio) (3.4)
Requirement already satisfied: sniffio in /usr/local/lib/python3.10/dist-packages (from httpx->gradio) (1.3.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->gradio) (1.0.7)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->gradio) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->gradio) (4.39.3)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->gradio) (1.4.4)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->gradio) (3.0.9)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->gradio) (1.26.15
Requirement already satisfied: anyio<5.0,>=3.0 in /usr/local/lib/python3.10/dist-packages (from httpcore<0.18.0,>=0.15.0->httpx-
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in /usr/local/lib/python3.10/dist-packages (from js
Collecting uc-micro-py (from linkify-it-py<3,>=1->markdown-it-py[linkify]>=2.0.0->gradio)
  Downloading uc_micro_py-1.0.2-py3-none-any.whl (6.2 kB)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.1->pandas->gradio)
Building wheels for collected packages: ffmpy
  Building wheel for ffmpy (setup.py) ... done
```

```
import gradio as gr
```

```
import numpy as np
import tensorflow as tf
from keras.models import load_model, save_model
from tensorflow.keras.preprocessing import image
import gradio as gr

# Define the model architecture
def create_model():
    # ... Define your model architecture here ...
    return model

# Load the trained model or create a new one
try:
    model = load_model('saved_model.h5')
except:
    model = create_model()
    # ... Train your model or load pre-trained weights here ...
    save_model(model, 'saved_model.h5')

# Define the pneumonia prediction function
def pneumoniaPrediction(img):
    img_array = image.img_to_array(img)
    img_array = img_array / 255.0
    img_batch = np.expand_dims(img_array, axis=0)
    preds = model.predict(img_batch)
    if preds[0][0] > 0.5:
        return "The Person is affected by Pneumonia"
    else:
        return "The Person is in Normal Condition"

# Define the Gradio interface
img = gr.inputs.Image(shape=(150, 150))
label = gr.outputs.Label()
```

```
# Customize the interface appearance
iface = gr.Interface(fn=pneumoniaPrediction,
                     inputs=img,
                     outputs=label,
                     title="Pneumonia Detection using Chest X-Ray",
                     theme="light",  # Choose from "default", "light", "dark", "huggingface"
                     layout="vertical",  # Choose from "vertical", "horizontal", "unaligned"
                     description="Upload an image of a chest X-ray to predict if the person has pneumonia.")

# Launch the Gradio interface
iface.launch(debug=True)
```

```
/usr/local/lib/python3.10/dist-packages/gradio/inputs.py:259: UserWarning: Usage of gradio.inputs is deprecated, and will not be su
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/gradio/inputs.py:262: UserWarning: `optional` parameter is deprecated, and it has no effect
  super().__init__(
/usr/local/lib/python3.10/dist-packages/gradio/outputs.py:197: UserWarning: Usage of gradio.outputs is deprecated, and will not be
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/gradio/outputs.py:200: UserWarning: The 'type' parameter has been deprecated. Use the Numbe
  super().__init__(num_top_classes=num_top_classes, type=type, label=label)
/usr/local/lib/python3.10/dist-packages/gradio/blocks.py:680: UserWarning: Cannot load light. Caught Exception: The space light doe
  warnings.warn(f"Cannot load {theme}. Caught Exception: {str(e)}")
<ipython-input-29-6584bc1f8ebb>:36: UserWarning: `layout` parameter is deprecated, and it has no effect
  iface = gr.Interface(fn=pneumoniaPrediction,
Colab notebook detected. This cell will run indefinitely so that you can see errors and logs. To turn off, set debug=False in launc
Note: opening Chrome Inspector may crash demo inside Colab notebooks.

To create a public link, set `share=True` in `launch()`.
Running on https://localhost:7860/
```




```
Keyboard interruption in main thread... closing server.
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
!mv * /content/drive/MyDrive/Your_Project_Folder/
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
from google.colab import files

# Download a file
files.download('pneumonia')
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-31-1724b37fc863> in <cell line: 4>()
      2
      3 # Download a file
----> 4 files.download('pneumonia')

/usr/local/lib/python3.10/dist-packages/google/colab/files.py in download(filename)
    220   if not _os.path.exists(filename):
    221     msg = 'Cannot find file: {}'.format(filename)
--> 222     raise FileNotFoundError(msg)  # pylint: disable=undefined-variable
    223
    224   comm_manager = _IPython.get_ipython().kernel.comm_manager

FileNotFoundError: Cannot find file: pneumonia
```

SEARCH STACK OVERFLOW

0s completed at 09:13