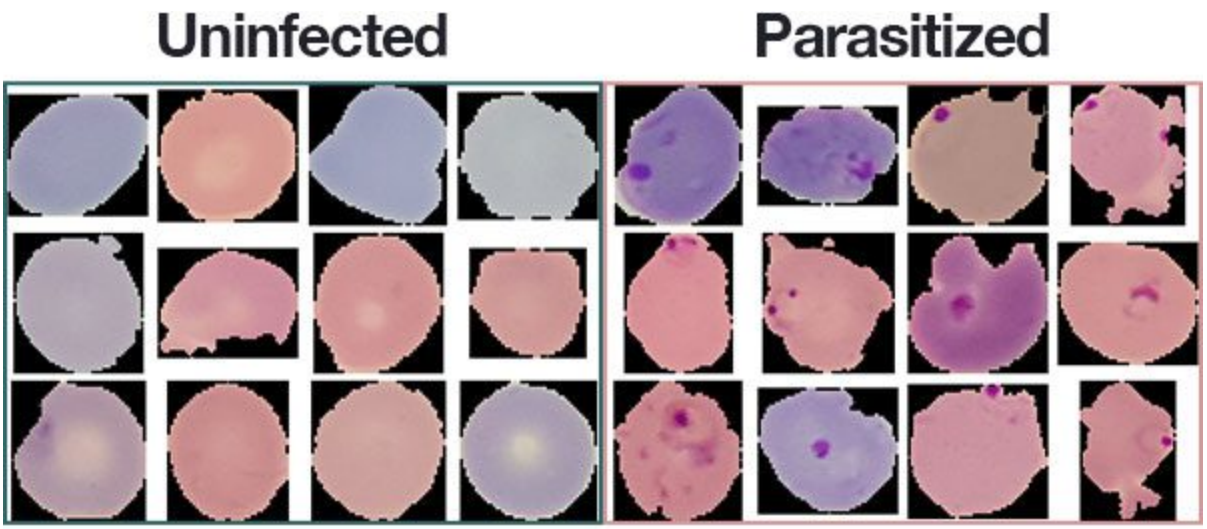


CNN With Custom Images: Malaria Cell Images Dataset

The dataset contains 2 folders - Infected - Uninfected

And a total of 27,558 images.

This Dataset was taken from the official NIH Website: https://ceb.nlm.nih.gov/repositories/malaria-datasets/

In [2]:
```
import os
import pandas as pd
import tensorflow as tf
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.image import imread
```

In [4]:
```
my_data= '/Users/usameturker/Desktop/DS/DL/cell_images/'
```

In [5]:
```
os.listdir(my_data)
```

Out[5]:
```
['.DS_Store', 'test', 'train']
```

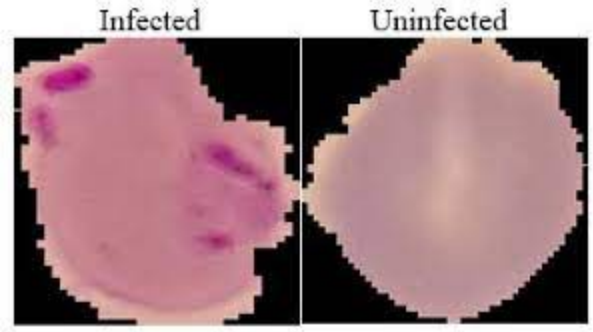
In [6]:
```
train_path = my_data+'/train/'
test_path = my_data+'/test/'
```

In [7]:
```
os.listdir(train_path)
```

Out[7]:
```
['.DS_Store', 'parasitized', 'uninfected']
```

In [8]:
```
os.listdir(test_path)
```

Out[8]:
```
['.DS_Store', 'parasitized', 'uninfected']
```



In [9]:
```
os.listdir(train_path+'parasitized')[:30]
```

Out[9]:
```
['C189P150ThinF_IMG_20151203_142224_cell_84.png',
 'C91P52ThinF_IMG_20150821_123116_cell_189.png',
 'C84P45ThinF_IMG_20150818_101226_cell_98.png',
 'C144P105ThinF_IMG_20151015_163432_cell_310.png',
 'C136P97ThinF_IMG_20151005_141803_cell_131.png',
 'C60P21thinF_IMG_20150804_105034_cell_105.png',
 'C176P137NThinF_IMG_20151201_122708_cell_126.png',
 'C136P97ThinF_IMG_20151005_142437_cell_119.png',
 'C173P134NThinF_IMG_20151130_115733_cell_227.png',
 'C85P46ThinF_IMG_20150820_105154_cell_221.png',
 'C101P62ThinF_IMG_20150918_151507_cell_49.png',
 'C116P77ThinF_IMG_20150930_171844_cell_100.png',
 'C48P9thinF_IMG_20150721_160406_cell_247.png',
 'C184P145ThinF_IMG_20151203_103114_cell_149.png',
 'C99P60ThinF_IMG_20150918_141620_cell_56.png',
 'C132P93ThinF_IMG_20151004_152045_cell_111.png',
 'C68P29N_ThinF_IMG_20150819_134112_cell_153.png',
 'C116P77ThinF_IMG_20150930_171844_cell_114.png',
 'C171P132ThinF_IMG_20151119_153150_cell_222.png',
 'C176P137NThinF_IMG_20151201_122708_cell_132.png',
 'C175P136NThinF_IMG_20151127_141325_cell_219.png',
 'C132P93ThinF_IMG_20151004_151733_cell_140.png',
 'C173P134NThinF_IMG_20151130_125501_cell_255.png',
 'C60P21thinF_IMG_20150804_105034_cell_111.png',
 'C118P79ThinF_IMG_20151002_105018_cell_144.png',
 'C51AP12thinF_IMG_20150724_153313_cell_109.png',
 'C189P150ThinF_IMG_20151203_142224_cell_90.png',
 'C144P105ThinF_IMG_20151015_163432_cell_304.png',
 'C136P97ThinF_IMG_20151005_141803_cell_125.png',
 'C65P26N_ThinF_IMG_20150818_154010_cell_200.png']
```

In [10]:
```
os.listdir(train_path+'parasitized')[17]
```

Out[10]:
```
'C116P77ThinF_IMG_20150930_171844_cell_114.png'
```

In [11]:
```
para_cell = train_path+'parasitized'+'/C100P61ThinF_IMG_20150918_144823_cell_158.png'
```

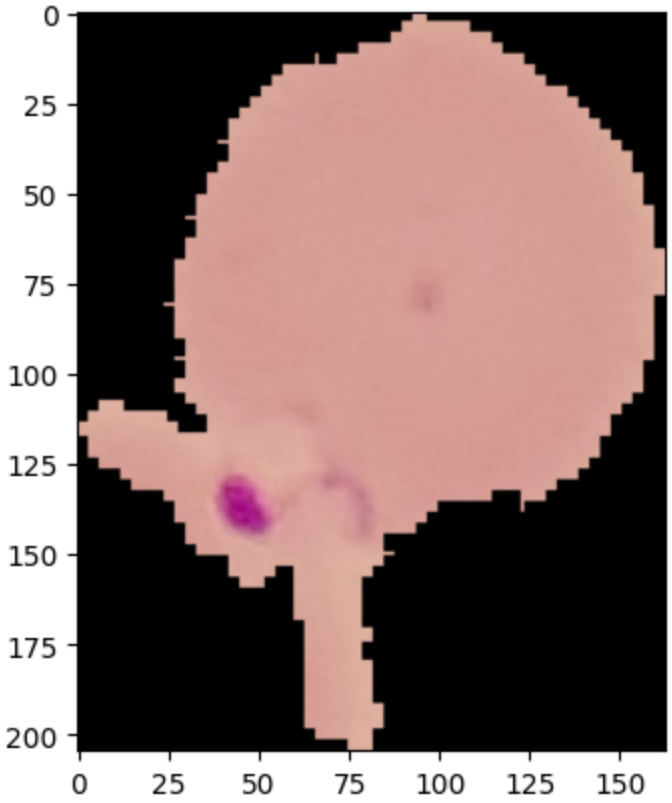
In [12]:
```
imread(para_cell)
```

Out[12]:
```
array([[[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       ...,

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]]], dtype=float32)
```

In [13]:
```
para_img= imread(para_cell)
```

In [14]:
```
plt.imshow(para_img)
```

Out[14]:
```
<matplotlib.image.AxesImage at 0x282da2890>
```



In [15]:
```
para_img.shape
```

Out[15]:
```
(205, 163, 3)
```

In [16]:
```
os.listdir(train_path+'uninfected')[:20]
```

Out[16]:
```
['C13NThinF_IMG_20150614_131318_cell_179.png',
 'C205ThinF_IMG_20151106_152003_cell_21.png',
 'C230ThinF_IMG_20151112_150647_cell_1.png',
 'C134P95ThinF_IMG_20151005_121834_cell_65.png',
 'C141P102ThinF_IMG_20151005_214836_cell_132.png',
 'C182P143NThinF_IMG_20151201_171950_cell_93.png',
 'C181P142ThinF_IMG_20151127_160616_cell_54.png',
 'C142P103ThinF_IMG_20151005_223257_cell_18.png',
 'C80P41ThinF_IMG_20150817_110608_cell_62.png',
 'C144P105ThinF_IMG_20151015_163142_cell_66.png',
 'C12NThinF_IMG_20150614_124212_cell_187.png',
 'C223ThinF_IMG_20151112_104708_cell_190.png',
 'C143P104ThinF_IMG_20151005_225746_cell_133.png',
 'C84P45ThinF_IMG_20150818_101056_cell_87.png',
 'C222ThinF_IMG_20151115_150925_cell_127.png',
 'C1_thinF_IMG_20150604_104919_cell_82.png',
 'C150P111ThinF_IMG_20151115_115950_cell_53.png',
 'C170P131ThinF_IMG_20151119_120233_cell_183.png',
 'C13NThinF_IMG_20150614_131529_cell_168.png',
 'C1_thinF_IMG_20150604_104919_cell_96.png']
```

In [17]:
```
uninfected_cell_path = train_path+'uninfected/'+os.listdir(train_path+'uninfected')[7]
```

In [18]:
```
uninfected_cell_path
```

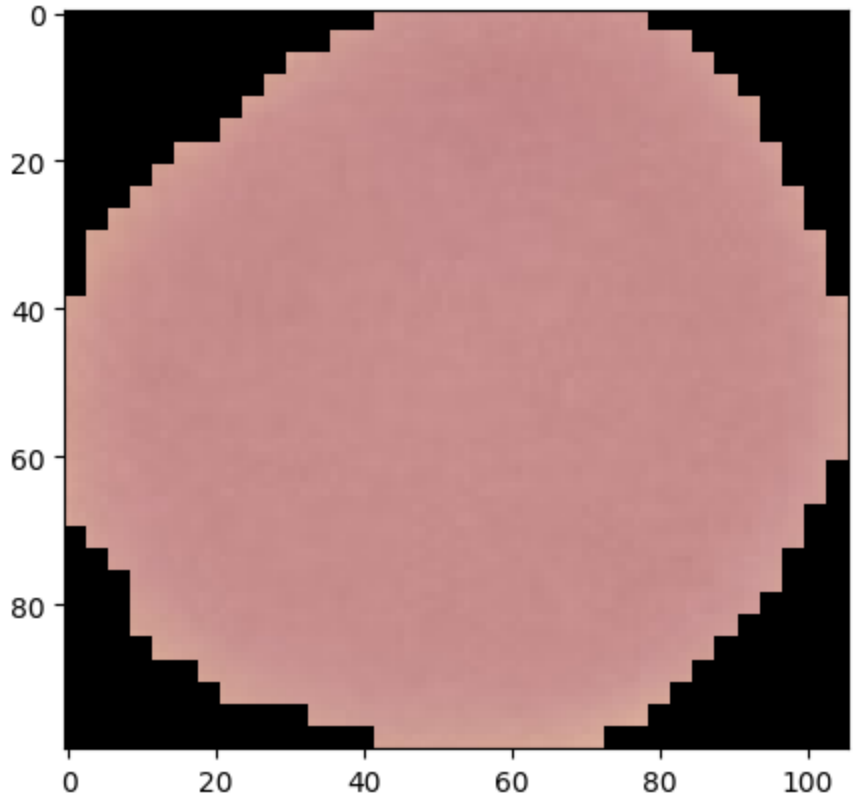
Out[18]:
```
'/Users/usameturker/Desktop/DS/DL/cell_images//train/uninfected/C142P103ThinF_IMG_20151005_223257_cell_18.png'
```

In [19]:
```
imread(uninfected_cell_path)
```

Out[19]:
```
array([[[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       ...,

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.],
        ...,
        [0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]]], dtype=float32)
```

In [20]:
```
uninfected_cell = imread(uninfected_cell_path)
plt.imshow(uninfected_cell)
```

Out[20]:
```
<matplotlib.image.AxesImage at 0x282e9b220>
```



In [21]:
```
len(os.listdir(train_path+'parasitized'))
```

Out[21]: 12480

In [22]:
```
len(os.listdir(train_path+'uninfected'))
```

Out[22]: 12480

In [23]:
```
uninfected_cell.shape
```

Out[23]: (100, 106, 3)

In [24]:
```
para_img.shape
```

Out[24]: (205, 163, 3)

In [25]:
```
x = []
y = []
for image in os.listdir(test_path+'uninfected'):

    img = imread(test_path+'uninfected/'+image)
    d1,d2,_ = img.shape
    x.append(d1)
    y.append(d2)
```

In [26]:
```
y[:20]
```

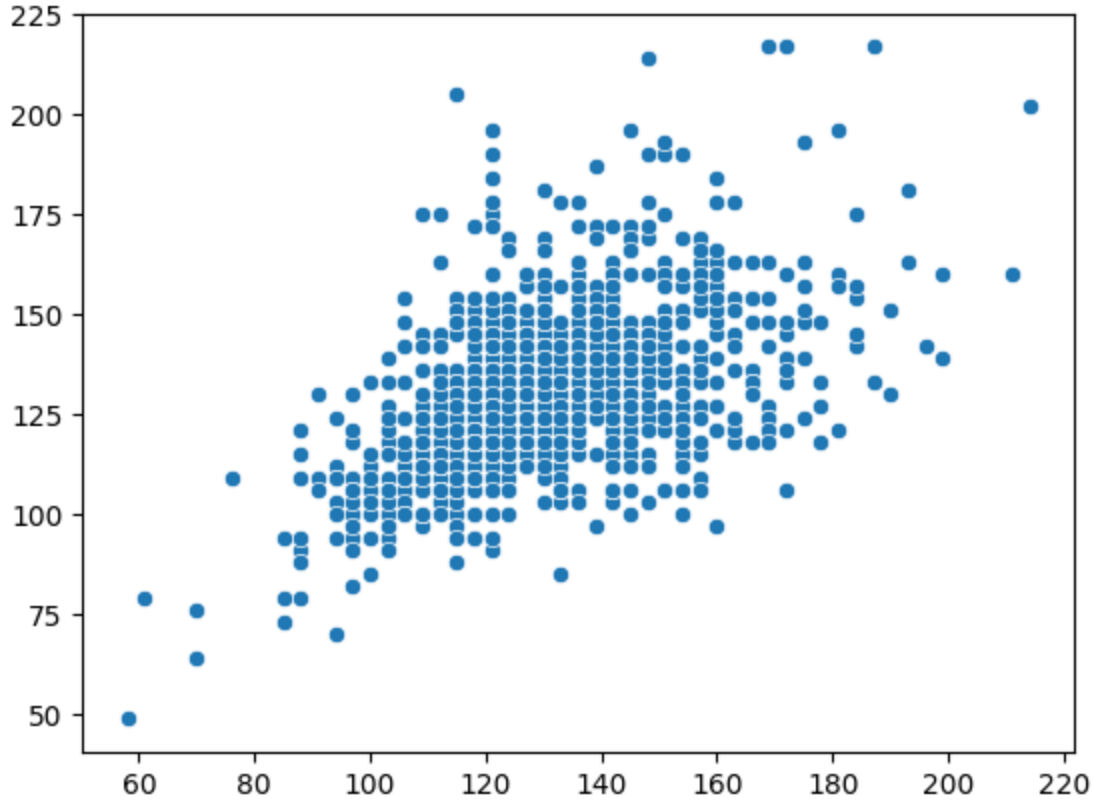
Out[26]:
```
[127,
 121,
 151,
 130,
 124,
 127,
 136,
 124,
 124,
 130,
 127,
 115,
 154,
 115,
 100,
 121,
 151,
 142,
 118,
 172]
```

In [27]:
```
x[:20]
```

Out[27]:
```
[127,
 109,
 127,
 118,
 127,
 118,
 133,
 115,
 133,
 142,
 139,
 121,
 124,
 133,
 103,
 148,
 130,
 118,
 127,
 142]
```

In [29]:
```
sns.scatterplot(x=x,y=y)
```

Out[29]:
```
<Axes: >
```



In [30]:
```
np.mean(x)
```

Out[30]: 130.92538461538462

In [31]:
```
np.mean(y)
```

Out[31]: 130.75

In [32]:
```
image_shape = (130,130,3)
```

Preparing the Data for the model

In [33]:
```
uninfected_cell.max()
```

Out[33]: 0.85490197

In [34]:
```
para_img.max()
```

Out[34]: 0.8980392

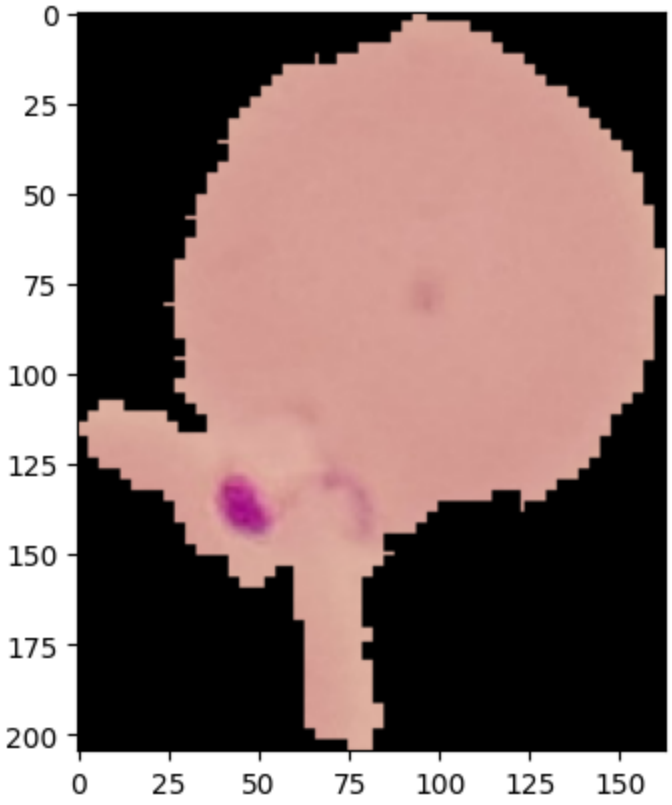
In [35]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

In [93]:
```
#help(ImageDataGenerator)
```

In [36]:
```
image_gen = ImageDataGenerator(rotation_range=15,
                               width_shift_range=0.10,
                               height_shift_range=0.10,
                               #rescale=1/255,
                               shear_range=0.1,
                               zoom_range=0.1,
                               horizontal_flip=True,
                               fill_mode='nearest')
```

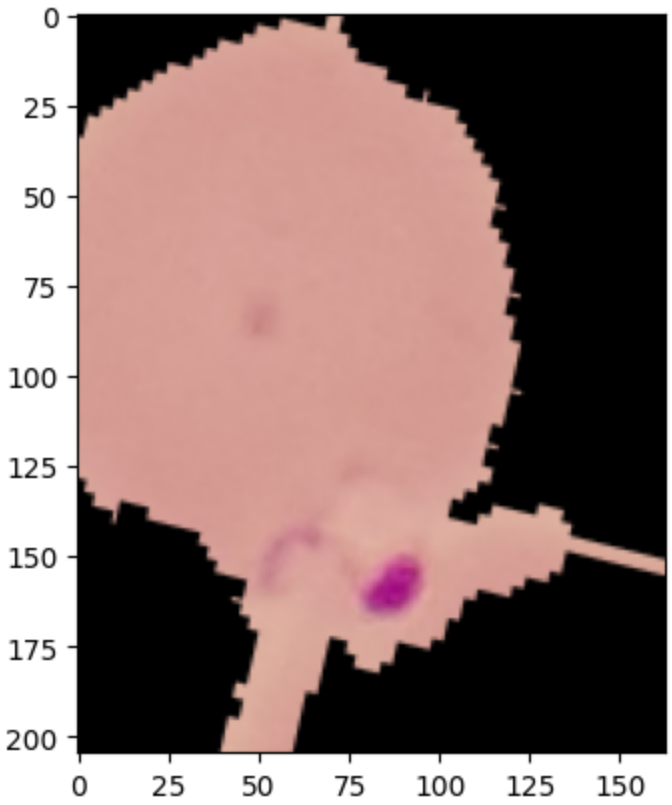
In [37]:
```
plt.imshow(para_img)
```

Out[37]:
```
<matplotlib.image.AxesImage at 0x2830f00a0>
```



In [38]:
```
plt.imshow(image_gen.random_transform(para_img))
```

Out[38]:
```
<matplotlib.image.AxesImage at 0x28444df00>
```



In [39]:
```
image_gen.flow_from_directory(train_path)
```

```
Found 24958 images belonging to 2 classes.
```

Out[39]:
```
<keras.preprocessing.image.DirectoryIterator at 0x2844d8d60>
```

In [40]:
```
image_gen.flow_from_directory(test_path)
```

```
Found 2600 images belonging to 2 classes.
```

Out[40]:
```
<keras.preprocessing.image.DirectoryIterator at 0x2844d9ab0>
```

Creating the Model

In [41]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Activation, Dropout, Flatten, Dense, Conv2D, MaxPool
```

In [42]:
```
model = Sequential()

model.add(Conv2D(filters=64, kernel_size=(3,3),input_shape=image_shape, padding='same',
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(filters=128, kernel_size=(3,3), padding='same', activation='relu',))
model.add(Conv2D(filters=128, kernel_size=(3,3), padding='same', activation='relu',))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(filters=256, kernel_size=(3,3), padding='same', activation='relu',))
model.add(Conv2D(filters=256, kernel_size=(3,3), padding='same', activation='relu',))
model.add(Conv2D(filters=256, kernel_size=(3,3), padding='same', activation='relu',))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(filters=128, kernel_size=(3,3), padding='same', activation='relu',))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(filters=64, kernel_size=(3,3), padding='same', activation='relu',))
model.add(MaxPooling2D(pool_size=(2, 2)))


model.add(Flatten())


model.add(Dense(128))
model.add(Activation('relu'))

model.add(Dropout(0.5))

model.add(Dense(1))
model.add(Activation('sigmoid'))

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

In [43]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 130, 130, 64)      1792

 max_pooling2d (MaxPooling2D  (None, 65, 65, 64)       0
 )

 conv2d_1 (Conv2D)           (None, 65, 65, 128)       73856

 conv2d_2 (Conv2D)           (None, 65, 65, 128)       147584

 max_pooling2d_1 (MaxPooling  (None, 32, 32, 128)      0
 2D)

 conv2d_3 (Conv2D)           (None, 32, 32, 256)       295168

 conv2d_4 (Conv2D)           (None, 32, 32, 256)       590080

 conv2d_5 (Conv2D)           (None, 32, 32, 256)       590080

 max_pooling2d_2 (MaxPooling  (None, 16, 16, 256)      0
 2D)

 conv2d_6 (Conv2D)           (None, 16, 16, 128)       295040

 max_pooling2d_3 (MaxPooling  (None, 8, 8, 128)        0
 2D)

 conv2d_7 (Conv2D)           (None, 8, 8, 64)          73792

 max_pooling2d_4 (MaxPooling  (None, 4, 4, 64)         0
 2D)

 flatten (Flatten)           (None, 1024)              0

 dense (Dense)               (None, 128)               131200

 activation (Activation)     (None, 128)               0

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 1)                 129

 activation_1 (Activation)   (None, 1)                 0

=================================================================
Total params: 2,198,721
Trainable params: 2,198,721
Non-trainable params: 0
_________________________________________________________________
```

Early Stopping

In [44]:
```
from tensorflow.keras.callbacks import EarlyStopping
```

In [45]:
```
early_stop = EarlyStopping(monitor='val_loss',patience=5)
```

Training the Model

In [46]:
```
batch_size = 32
train_image_gen = image_gen.flow_from_directory(train_path,
                                                target_size=(130,130),
                                                 color_mode='rgb',
                                                batch_size=batch_size,
                                                class_mode='binary', shuffle=True, seed=4
```

```
Found 24958 images belonging to 2 classes.
```

In [47]:
```
test_image_gen = image_gen.flow_from_directory(test_path,
                                                target_size=(130,130),
                                                color_mode='rgb',
                                                batch_size=batch_size,
                                                class_mode='binary', shuffle=True, seed=4
```

```
Found 2600 images belonging to 2 classes.
```

In [48]:
```
train_image_gen.class_indices
```

Out[48]:
```
{'parasitized': 0, 'uninfected': 1}
```

In [49]:
```
test_image_gen.class_indices
```

Out[49]:
```
{'parasitized': 0, 'uninfected': 1}
```

In [ ]:
```
results = model.fit(train_image_gen,epochs=20,
                    validation_data=test_image_gen,
                    callbacks=[early_stop])
```

```
Epoch 1/20
780/780 [==============================] - 1478s 2s/step - loss: 0.6961 - accuracy: 0.50
82 - val_loss: 0.6931 - val_accuracy: 0.5000
Epoch 2/20
780/780 [==============================] - 1458s 2s/step - loss: 0.6924 - accuracy: 0.51
93 - val_loss: 0.6942 - val_accuracy: 0.4719
Epoch 3/20
780/780 [==============================] - 1480s 2s/step - loss: 0.6925 - accuracy: 0.53
35 - val_loss: 0.6888 - val_accuracy: 0.5238
Epoch 4/20
780/780 [==============================] - 1710s 2s/step - loss: 0.6443 - accuracy: 0.60
68 - val_loss: 0.3118 - val_accuracy: 0.9088
Epoch 5/20
780/780 [==============================] - 1502s 2s/step - loss: 0.2005 - accuracy: 0.94
23 - val_loss: 0.1772 - val_accuracy: 0.9458
Epoch 6/20
780/780 [==============================] - 1507s 2s/step - loss: 0.1729 - accuracy: 0.94
86 - val_loss: 0.1724 - val_accuracy: 0.9496
Epoch 7/20
780/780 [==============================] - 1489s 2s/step - loss: 0.1604 - accuracy: 0.95
12 - val_loss: 0.1640 - val_accuracy: 0.9504
Epoch 8/20
780/780 [==============================] - 4497s 6s/step - loss: 0.1532 - accuracy: 0.95
26 - val_loss: 0.1560 - val_accuracy: 0.9504
Epoch 9/20
 95/780 [==>...........................] - ETA: 3:15:47 - loss: 0.1424 - accuracy: 0.959
5
```