data_aug_adam_val_95_28

October 7, 2021

1 Covid Classifier Model

1.0.1 Goals

Classify: - Normal CXR - Viral Pneumonia CXR - COVID CXR

1.1 Create Directories for Dataset

Separate the data to use later as generators.

[ ]:
```
# Matriz de confusion, cambiar learnings rates (learning rates dinamicos),␣
↪dropouts 0.3 & 0.2, Batch Normalization
# K-Fold o avg de modelos
import os

BASE_PATH = '/home/hivini/learn/research/new-covid'
ORIGINAL_DATASET_DIR = os.path.join(BASE_PATH, 'COVID-19_Radiography_Dataset')
ORIGINAL_VIRAL_DIR = os.path.join(ORIGINAL_DATASET_DIR, 'Viral Pneumonia')
ORIGINAL_COVID_DIR = os.path.join(ORIGINAL_DATASET_DIR, 'COVID')
ORIGINAL_NORMAL_DIR = os.path.join(ORIGINAL_DATASET_DIR, 'Normal')
DATASET_DIR = os.path.join(BASE_PATH, 'small_dataset')
TRAIN_DIR = os.path.join(DATASET_DIR, 'train')
VALIDATION_DIR = os.path.join(DATASET_DIR, 'validation')
TEST_DIR = os.path.join(DATASET_DIR, 'test')
TRAIN_VIRAL_DIR = os.path.join(TRAIN_DIR, 'viral_pneumonia')
TRAIN_COVID_DIR = os.path.join(TRAIN_DIR, 'covid')
TRAIN_NORMAL_DIR = os.path.join(TRAIN_DIR, 'normal')
VALIDATION_VIRAL_DIR = os.path.join(VALIDATION_DIR, 'viral_pneumonia')
VALIDATION_COVID_DIR = os.path.join(VALIDATION_DIR, 'covid')
VALIDATION_NORMAL_DIR = os.path.join(VALIDATION_DIR, 'normal')
TEST_VIRAL_DIR = os.path.join(TEST_DIR, 'viral_pneumonia')
TEST_COVID_DIR = os.path.join(TEST_DIR, 'covid')
TEST_NORMAL_DIR = os.path.join(TEST_DIR, 'normal')


def createDir(path: str) -> None:
    if not os.path.exists(path):
        os.mkdir(path)
```

```
createDir(DATASET_DIR)
createDir(TRAIN_DIR)
createDir(VALIDATION_DIR)
createDir(TEST_DIR)
createDir(TRAIN_VIRAL_DIR)
createDir(TRAIN_COVID_DIR)
createDir(TRAIN_NORMAL_DIR)
createDir(VALIDATION_VIRAL_DIR)
createDir(VALIDATION_COVID_DIR)
createDir(VALIDATION_NORMAL_DIR)
createDir(TEST_VIRAL_DIR)
createDir(TEST_COVID_DIR)
createDir(TEST_NORMAL_DIR)
```

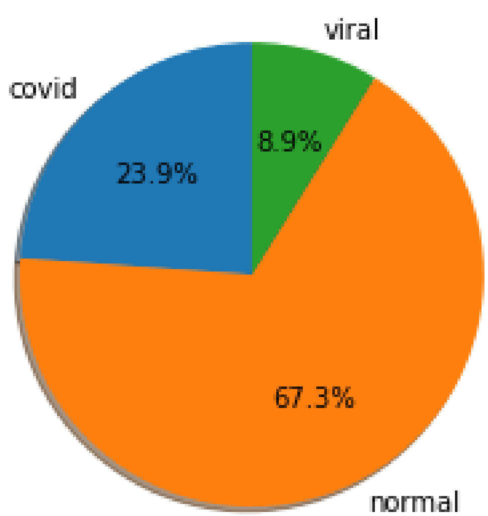
[ ]:
```
import numpy as np
import shutil


def generate_sets(source: str):
    allFiles = os.listdir(source)
    np.random.shuffle(allFiles)
    return np.split(np.array(allFiles), [int(len(allFiles)*0.7),
 ↪int(len(allFiles)*0.85)])


def saveAndSeparateFiles(src_dir: str, train_dir: str, val_dir: str, test_dir):
    train_fnames, val_fnames, test_fnames = generate_sets(src_dir)
    for fname in train_fnames:
        src = os.path.join(src_dir, fname)
        dst = os.path.join(train_dir, fname)
        shutil.copyfile(src, dst)

    for fname in val_fnames:
        src = os.path.join(src_dir, fname)
        dst = os.path.join(val_dir, fname)
        shutil.copyfile(src, dst)

    for fname in test_fnames:
        src = os.path.join(src_dir, fname)
        dst = os.path.join(test_dir, fname)
        shutil.copyfile(src, dst)

create = False
if create:
    saveAndSeparateFiles(ORIGINAL_NORMAL_DIR, TRAIN_NORMAL_DIR,
                         VALIDATION_NORMAL_DIR, TEST_NORMAL_DIR)
```

```
    saveAndSeparateFiles(ORIGINAL_COVID_DIR, TRAIN_COVID_DIR,
                         VALIDATION_COVID_DIR, TEST_COVID_DIR)
    saveAndSeparateFiles(ORIGINAL_VIRAL_DIR, TRAIN_VIRAL_DIR,
                         VALIDATION_VIRAL_DIR, TEST_VIRAL_DIR)
```

1.2 Counting our images

[ ]:
```
import tensorflow as tf
import matplotlib.pyplot as plt
normal_train = tf.io.gfile.glob(TRAIN_NORMAL_DIR + '/*')
viral_train = tf.io.gfile.glob(TRAIN_VIRAL_DIR + '/*')
covid_train = tf.io.gfile.glob(TRAIN_COVID_DIR + '/*')

# Plotting Distribution of Each Classes
image_count = {'covid': len(covid_train), 'normal': len(
    normal_train), 'viral': len(viral_train)}
fig1, ax1 = plt.subplots()
ax1.pie(image_count.values(),
        labels=image_count.keys(),
        shadow=True,
        autopct='%1.1f%%',
        startangle=90)
plt.show()
```

```
2021-10-07 20:14:24.777820: I
tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully
opened dynamic library libcudart.so.10.1
```



1.3 Create our Covnet Model

In this case we are doing a multi class classification, our total clases are 3: - Viral CXR - Covid CXR - Normal CXR

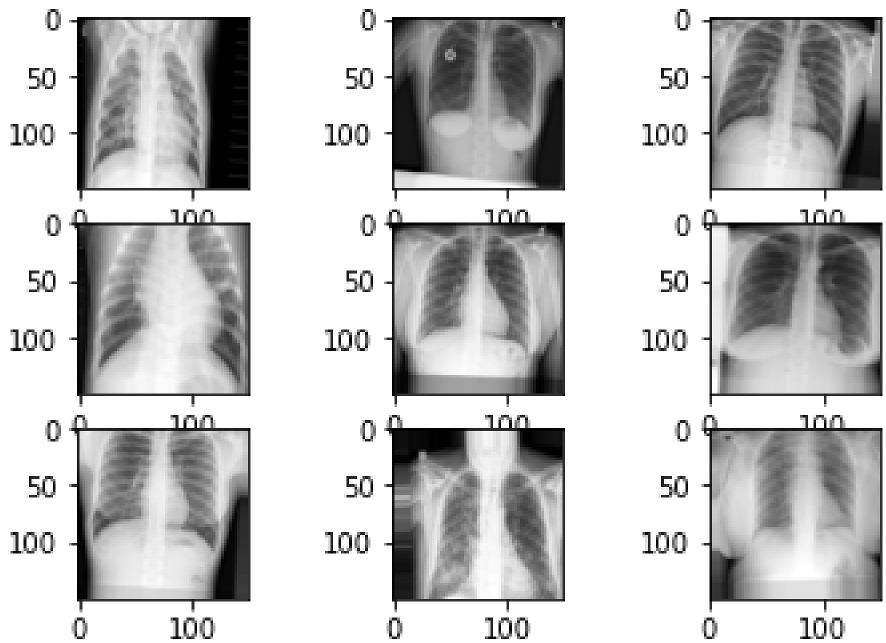
Our neural network will output neurons as 3 classes that will calculate the probability of being one using the softmax function.

[ ]:
```python
from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(
    rescale=1./255,
    featurewise_center=False,  # set input mean to 0 over the dataset
    samplewise_center=False,  # set each sample mean to 0
    featurewise_std_normalization=False,  # divide inputs by std of the dataset
    samplewise_std_normalization=False,  # divide each input by its std
    zca_whitening=False,  # apply ZCA whitening
    # randomly rotate images in the range (degrees, 0 to 180)
    rotation_range=10,
    zoom_range=0.1,  # Randomly zoom image
    # randomly shift images horizontally (fraction of total width)
    width_shift_range=0.1,
    # randomly shift images vertically (fraction of total height)
    height_shift_range=0.1,
    horizontal_flip=False,  # randomly flip images
    vertical_flip=False  # randomly flip images
)

# train_datagen = ImageDataGenerator(rescale=1./255)
test_datagen = ImageDataGenerator(rescale=1./255)
evaluate_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    TRAIN_DIR,
    target_size=(150, 150),
    batch_size=32,
    class_mode='categorical',
    color_mode='grayscale'
)

print(train_generator.class_indices)

validation_generator = test_datagen.flow_from_directory(
    VALIDATION_DIR,
    target_size=(150, 150),
    batch_size=32,
    class_mode='categorical',
    color_mode='grayscale'
```

```
)

print(validation_generator.class_indices)

test_generator = evaluate_datagen.flow_from_directory(
    TEST_DIR,
    target_size=(150, 150),
    batch_size=32,
    class_mode='categorical',
    color_mode='grayscale'
)

print(test_generator.class_indices)
```

```
Found 10606 images belonging to 3 classes.
{'covid': 0, 'normal': 1, 'viral_pneumonia': 2}
Found 2273 images belonging to 3 classes.
{'covid': 0, 'normal': 1, 'viral_pneumonia': 2}
Found 2274 images belonging to 3 classes.
{'covid': 0, 'normal': 1, 'viral_pneumonia': 2}
```

[ ]:
```
for X_batch, y_batch in train_generator:
        # create a grid of 3x3 images
        for i in range(0, 9):
                plt.subplot(330 + 1 + i)
                plt.imshow(X_batch[i].reshape(150, 150), cmap=plt.
 ↪get_cmap('gray'))
        # show the plot
        plt.show()
        break
```



[ ]:
```
from keras.layers import Conv2D, BatchNormalization, MaxPooling2D, Dropout,␣
 ↪Flatten, Dense
from keras.models import Sequential
from keras import backend

backend.clear_session()

model = Sequential()
model.add(Conv2D(64, (3, 3), activation='relu', input_shape=(150, 150, 1)))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dropout(0.5))
model.add(Dense(512, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(3, activation='softmax'))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 148, 148, 64)      640
_________________________________________________________________
batch_normalization (BatchNo (None, 148, 148, 64)      256
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 74, 74, 64)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 72, 72, 64)        36928
_________________________________________________________________
batch_normalization_1 (Batch (None, 72, 72, 64)        256
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 36, 36, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 34, 34, 128)       73856
_________________________________________________________________
batch_normalization_2 (Batch (None, 34, 34, 128)       512
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 17, 17, 128)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 15, 15, 128)       147584
_________________________________________________________________
batch_normalization_3 (Batch (None, 15, 15, 128)       512
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 7, 7, 128)         0
_________________________________________________________________
flatten (Flatten)            (None, 6272)              0
_________________________________________________________________
dropout (Dropout)            (None, 6272)              0
_________________________________________________________________
dense (Dense)                (None, 512)               3211776
_________________________________________________________________
dense_1 (Dense)              (None, 64)                32832
_________________________________________________________________
dense_2 (Dense)              (None, 3)                 195
=================================================================
Total params: 3,505,347
Trainable params: 3,504,579
Non-trainable params: 768
_________________________________________________________________
```

[ ]:
```
from keras import optimizers
```

```
# opt = RMSprop(lr=0.0001, decay=1e-6)
opt = optimizers.Adam(learning_rate=1e-5, decay=1e-7)

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

[ ]:
```
import numpy as np
from sklearn.utils import class_weight
from keras.callbacks import EarlyStopping
from keras.callbacks import ModelCheckpoint

classes = train_generator.classes
class_weights = class_weight.compute_class_weight(None,
                                                  np.unique(classes),
                                                  classes)
best_model_path = os.path.join(BASE_PATH, 'best_model.h5')
es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=8)
mc = ModelCheckpoint(best_model_path, monitor='val_accuracy', mode='max',␣
↪verbose=1, save_best_only=True)
history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.n // 32,
    epochs=150,
    validation_data=validation_generator,
    class_weight=dict(zip(np.unique(classes), class_weights)),
    callbacks=[es, mc]
)
```

```
Epoch 1/150
331/331 [==============================] - 36s 108ms/step - loss: 1.1811 -
accuracy: 0.5959 - val_loss: 1.6331 - val_accuracy: 0.2411

Epoch 00001: val_accuracy improved from -inf to 0.24109, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 2/150
331/331 [==============================] - 33s 101ms/step - loss: 0.6353 -
accuracy: 0.7354 - val_loss: 0.4355 - val_accuracy: 0.8170

Epoch 00002: val_accuracy improved from 0.24109 to 0.81698, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 3/150
331/331 [==============================] - 32s 97ms/step - loss: 0.5500 -
accuracy: 0.7627 - val_loss: 0.3936 - val_accuracy: 0.8214

Epoch 00003: val_accuracy improved from 0.81698 to 0.82138, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 4/150
```

```
331/331 [==============================] - 32s 97ms/step - loss: 0.4964 -
accuracy: 0.7868 - val_loss: 0.3578 - val_accuracy: 0.8460

Epoch 00004: val_accuracy improved from 0.82138 to 0.84602, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 5/150
331/331 [==============================] - 33s 98ms/step - loss: 0.4639 -
accuracy: 0.8095 - val_loss: 0.3470 - val_accuracy: 0.8482

Epoch 00005: val_accuracy improved from 0.84602 to 0.84822, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 6/150
331/331 [==============================] - 32s 97ms/step - loss: 0.4297 -
accuracy: 0.8165 - val_loss: 0.3197 - val_accuracy: 0.8614

Epoch 00006: val_accuracy improved from 0.84822 to 0.86142, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 7/150
331/331 [==============================] - 32s 98ms/step - loss: 0.4217 -
accuracy: 0.8239 - val_loss: 0.3027 - val_accuracy: 0.8667

Epoch 00007: val_accuracy improved from 0.86142 to 0.86670, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 8/150
331/331 [==============================] - 32s 98ms/step - loss: 0.3966 -
accuracy: 0.8354 - val_loss: 0.2895 - val_accuracy: 0.8759

Epoch 00008: val_accuracy improved from 0.86670 to 0.87593, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 9/150
331/331 [==============================] - 32s 97ms/step - loss: 0.3773 -
accuracy: 0.8380 - val_loss: 0.2666 - val_accuracy: 0.8852

Epoch 00009: val_accuracy improved from 0.87593 to 0.88517, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 10/150
331/331 [==============================] - 32s 97ms/step - loss: 0.3625 -
accuracy: 0.8536 - val_loss: 0.2635 - val_accuracy: 0.8861

Epoch 00010: val_accuracy improved from 0.88517 to 0.88605, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 11/150
331/331 [==============================] - 32s 98ms/step - loss: 0.3531 -
accuracy: 0.8498 - val_loss: 0.2518 - val_accuracy: 0.8962

Epoch 00011: val_accuracy improved from 0.88605 to 0.89617, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 12/150
```

```
331/331 [==============================] - 32s 98ms/step - loss: 0.3289 -
accuracy: 0.8685 - val_loss: 0.2681 - val_accuracy: 0.8755

Epoch 00012: val_accuracy did not improve from 0.89617
Epoch 13/150
331/331 [==============================] - 33s 99ms/step - loss: 0.3219 -
accuracy: 0.8657 - val_loss: 0.2387 - val_accuracy: 0.9001

Epoch 00013: val_accuracy improved from 0.89617 to 0.90013, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 14/150
331/331 [==============================] - 32s 98ms/step - loss: 0.3104 -
accuracy: 0.8723 - val_loss: 0.2520 - val_accuracy: 0.8869

Epoch 00014: val_accuracy did not improve from 0.90013
Epoch 15/150
331/331 [==============================] - 32s 98ms/step - loss: 0.2965 -
accuracy: 0.8830 - val_loss: 0.2434 - val_accuracy: 0.8971

Epoch 00015: val_accuracy did not improve from 0.90013
Epoch 16/150
331/331 [==============================] - 34s 101ms/step - loss: 0.2969 -
accuracy: 0.8806 - val_loss: 0.2118 - val_accuracy: 0.9147

Epoch 00016: val_accuracy improved from 0.90013 to 0.91465, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 17/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2868 -
accuracy: 0.8831 - val_loss: 0.2367 - val_accuracy: 0.9028

Epoch 00017: val_accuracy did not improve from 0.91465
Epoch 18/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2923 -
accuracy: 0.8842 - val_loss: 0.2188 - val_accuracy: 0.9138

Epoch 00018: val_accuracy did not improve from 0.91465
Epoch 19/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2632 -
accuracy: 0.8945 - val_loss: 0.2025 - val_accuracy: 0.9243

Epoch 00019: val_accuracy improved from 0.91465 to 0.92433, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 20/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2504 -
accuracy: 0.8963 - val_loss: 0.2460 - val_accuracy: 0.9001

Epoch 00020: val_accuracy did not improve from 0.92433
Epoch 21/150
```

```
331/331 [==============================] - 33s 100ms/step - loss: 0.2438 -
accuracy: 0.9013 - val_loss: 0.2119 - val_accuracy: 0.9173

Epoch 00021: val_accuracy did not improve from 0.92433
Epoch 22/150
331/331 [==============================] - 39s 117ms/step - loss: 0.2328 -
accuracy: 0.9100 - val_loss: 0.2993 - val_accuracy: 0.8869

Epoch 00022: val_accuracy did not improve from 0.92433
Epoch 23/150
331/331 [==============================] - 33s 101ms/step - loss: 0.2528 -
accuracy: 0.9013 - val_loss: 0.2243 - val_accuracy: 0.9081

Epoch 00023: val_accuracy did not improve from 0.92433
Epoch 24/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2335 -
accuracy: 0.9075 - val_loss: 0.1725 - val_accuracy: 0.9344

Epoch 00024: val_accuracy improved from 0.92433 to 0.93445, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 25/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2201 -
accuracy: 0.9155 - val_loss: 0.2458 - val_accuracy: 0.9045

Epoch 00025: val_accuracy did not improve from 0.93445
Epoch 26/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2123 -
accuracy: 0.9144 - val_loss: 0.1720 - val_accuracy: 0.9340

Epoch 00026: val_accuracy did not improve from 0.93445
Epoch 27/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2226 -
accuracy: 0.9140 - val_loss: 0.2461 - val_accuracy: 0.9028

Epoch 00027: val_accuracy did not improve from 0.93445
Epoch 28/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2106 -
accuracy: 0.9187 - val_loss: 0.1939 - val_accuracy: 0.9278

Epoch 00028: val_accuracy did not improve from 0.93445
Epoch 29/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2133 -
accuracy: 0.9168 - val_loss: 0.1702 - val_accuracy: 0.9362

Epoch 00029: val_accuracy improved from 0.93445 to 0.93621, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 30/150
331/331 [==============================] - 32s 97ms/step - loss: 0.2060 -
```

```
accuracy: 0.9200 - val_loss: 0.2272 - val_accuracy: 0.9168

Epoch 00030: val_accuracy did not improve from 0.93621
Epoch 31/150
331/331 [==============================] - 33s 100ms/step - loss: 0.1989 -
accuracy: 0.9221 - val_loss: 0.1639 - val_accuracy: 0.9419

Epoch 00031: val_accuracy improved from 0.93621 to 0.94193, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 32/150
331/331 [==============================] - 32s 98ms/step - loss: 0.2006 -
accuracy: 0.9201 - val_loss: 0.1861 - val_accuracy: 0.9296

Epoch 00032: val_accuracy did not improve from 0.94193
Epoch 33/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1900 -
accuracy: 0.9280 - val_loss: 0.1598 - val_accuracy: 0.9393

Epoch 00033: val_accuracy did not improve from 0.94193
Epoch 34/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1854 -
accuracy: 0.9290 - val_loss: 0.1560 - val_accuracy: 0.9415

Epoch 00034: val_accuracy did not improve from 0.94193
Epoch 35/150
331/331 [==============================] - 32s 97ms/step - loss: 0.1928 -
accuracy: 0.9256 - val_loss: 0.1750 - val_accuracy: 0.9362

Epoch 00035: val_accuracy did not improve from 0.94193
Epoch 36/150
331/331 [==============================] - 32s 97ms/step - loss: 0.1756 -
accuracy: 0.9320 - val_loss: 0.1888 - val_accuracy: 0.9300

Epoch 00036: val_accuracy did not improve from 0.94193
Epoch 37/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1704 -
accuracy: 0.9343 - val_loss: 0.1431 - val_accuracy: 0.9485

Epoch 00037: val_accuracy improved from 0.94193 to 0.94853, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 38/150
331/331 [==============================] - 32s 97ms/step - loss: 0.1706 -
accuracy: 0.9365 - val_loss: 0.2093 - val_accuracy: 0.9182

Epoch 00038: val_accuracy did not improve from 0.94853
Epoch 39/150
331/331 [==============================] - 33s 99ms/step - loss: 0.1698 -
accuracy: 0.9343 - val_loss: 0.1483 - val_accuracy: 0.9459
```

```
Epoch 00039: val_accuracy did not improve from 0.94853
Epoch 40/150
331/331 [==============================] - 33s 99ms/step - loss: 0.1831 -
accuracy: 0.9300 - val_loss: 0.3303 - val_accuracy: 0.8808

Epoch 00040: val_accuracy did not improve from 0.94853
Epoch 41/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1693 -
accuracy: 0.9342 - val_loss: 0.1278 - val_accuracy: 0.9538

Epoch 00041: val_accuracy improved from 0.94853 to 0.95381, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 42/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1604 -
accuracy: 0.9406 - val_loss: 0.1282 - val_accuracy: 0.9529

Epoch 00042: val_accuracy did not improve from 0.95381
Epoch 43/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1759 -
accuracy: 0.9362 - val_loss: 0.1600 - val_accuracy: 0.9459

Epoch 00043: val_accuracy did not improve from 0.95381
Epoch 44/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1678 -
accuracy: 0.9350 - val_loss: 0.1712 - val_accuracy: 0.9388

Epoch 00044: val_accuracy did not improve from 0.95381
Epoch 45/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1472 -
accuracy: 0.9466 - val_loss: 0.2125 - val_accuracy: 0.9221

Epoch 00045: val_accuracy did not improve from 0.95381
Epoch 46/150
331/331 [==============================] - 34s 104ms/step - loss: 0.1656 -
accuracy: 0.9405 - val_loss: 0.1405 - val_accuracy: 0.9463

Epoch 00046: val_accuracy did not improve from 0.95381
Epoch 47/150
331/331 [==============================] - 38s 114ms/step - loss: 0.1587 -
accuracy: 0.9422 - val_loss: 0.1406 - val_accuracy: 0.9454

Epoch 00047: val_accuracy did not improve from 0.95381
Epoch 48/150
331/331 [==============================] - 32s 97ms/step - loss: 0.1546 -
accuracy: 0.9406 - val_loss: 0.1440 - val_accuracy: 0.9415

Epoch 00048: val_accuracy did not improve from 0.95381
```

```
Epoch 49/150
331/331 [==============================] - 33s 98ms/step - loss: 0.1411 -
accuracy: 0.9506 - val_loss: 0.1244 - val_accuracy: 0.9520

Epoch 00049: val_accuracy did not improve from 0.95381
Epoch 50/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1466 -
accuracy: 0.9471 - val_loss: 0.1227 - val_accuracy: 0.9542

Epoch 00050: val_accuracy improved from 0.95381 to 0.95425, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 51/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1399 -
accuracy: 0.9502 - val_loss: 0.1145 - val_accuracy: 0.9604

Epoch 00051: val_accuracy improved from 0.95425 to 0.96040, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 52/150
331/331 [==============================] - 33s 98ms/step - loss: 0.1493 -
accuracy: 0.9454 - val_loss: 0.1282 - val_accuracy: 0.9534

Epoch 00052: val_accuracy did not improve from 0.96040
Epoch 53/150
331/331 [==============================] - 32s 97ms/step - loss: 0.1451 -
accuracy: 0.9471 - val_loss: 0.1304 - val_accuracy: 0.9525

Epoch 00053: val_accuracy did not improve from 0.96040
Epoch 54/150
331/331 [==============================] - 32s 97ms/step - loss: 0.1453 -
accuracy: 0.9455 - val_loss: 0.1264 - val_accuracy: 0.9534

Epoch 00054: val_accuracy did not improve from 0.96040
Epoch 55/150
331/331 [==============================] - 33s 99ms/step - loss: 0.1311 -
accuracy: 0.9511 - val_loss: 0.1231 - val_accuracy: 0.9556

Epoch 00055: val_accuracy did not improve from 0.96040
Epoch 56/150
331/331 [==============================] - 33s 100ms/step - loss: 0.1331 -
accuracy: 0.9511 - val_loss: 0.1078 - val_accuracy: 0.9630

Epoch 00056: val_accuracy improved from 0.96040 to 0.96304, saving model to
/home/hivini/learn/research/new-covid/best_model.h5
Epoch 57/150
331/331 [==============================] - 33s 100ms/step - loss: 0.1352 -
accuracy: 0.9520 - val_loss: 0.1473 - val_accuracy: 0.9424

Epoch 00057: val_accuracy did not improve from 0.96304
```

```
Epoch 58/150
331/331 [==============================] - 34s 102ms/step - loss: 0.1304 -
accuracy: 0.9513 - val_loss: 0.3040 - val_accuracy: 0.8883

Epoch 00058: val_accuracy did not improve from 0.96304
Epoch 59/150
331/331 [==============================] - 33s 100ms/step - loss: 0.1319 -
accuracy: 0.9511 - val_loss: 0.1260 - val_accuracy: 0.9551

Epoch 00059: val_accuracy did not improve from 0.96304
Epoch 60/150
331/331 [==============================] - 33s 99ms/step - loss: 0.1343 -
accuracy: 0.9503 - val_loss: 0.2046 - val_accuracy: 0.9212

Epoch 00060: val_accuracy did not improve from 0.96304
Epoch 61/150
331/331 [==============================] - 33s 100ms/step - loss: 0.1285 -
accuracy: 0.9499 - val_loss: 0.1233 - val_accuracy: 0.9534

Epoch 00061: val_accuracy did not improve from 0.96304
Epoch 62/150
331/331 [==============================] - 33s 98ms/step - loss: 0.1302 -
accuracy: 0.9519 - val_loss: 0.1722 - val_accuracy: 0.9305

Epoch 00062: val_accuracy did not improve from 0.96304
Epoch 63/150
331/331 [==============================] - 32s 98ms/step - loss: 0.1256 -
accuracy: 0.9552 - val_loss: 0.1041 - val_accuracy: 0.9626

Epoch 00063: val_accuracy did not improve from 0.96304
Epoch 64/150
331/331 [==============================] - 33s 98ms/step - loss: 0.1235 -
accuracy: 0.9542 - val_loss: 0.1200 - val_accuracy: 0.9573

Epoch 00064: val_accuracy did not improve from 0.96304
Epoch 65/150
331/331 [==============================] - 32s 97ms/step - loss: 0.1157 -
accuracy: 0.9580 - val_loss: 0.1047 - val_accuracy: 0.9604

Epoch 00065: val_accuracy did not improve from 0.96304
Epoch 66/150
331/331 [==============================] - 32s 97ms/step - loss: 0.1067 -
accuracy: 0.9613 - val_loss: 0.1170 - val_accuracy: 0.9564

Epoch 00066: val_accuracy did not improve from 0.96304
Epoch 67/150
331/331 [==============================] - 32s 97ms/step - loss: 0.1211 -
accuracy: 0.9561 - val_loss: 0.1134 - val_accuracy: 0.9613
```

```
Epoch 00067: val_accuracy did not improve from 0.96304
Epoch 68/150
331/331 [==============================] - 32s 97ms/step - loss: 0.1259 -
accuracy: 0.9533 - val_loss: 0.1126 - val_accuracy: 0.9591

Epoch 00068: val_accuracy did not improve from 0.96304
Epoch 69/150
331/331 [==============================] - 33s 100ms/step - loss: 0.1228 -
accuracy: 0.9567 - val_loss: 0.1229 - val_accuracy: 0.9542

Epoch 00069: val_accuracy did not improve from 0.96304
Epoch 70/150
331/331 [==============================] - 34s 101ms/step - loss: 0.1139 -
accuracy: 0.9597 - val_loss: 0.1443 - val_accuracy: 0.9450

Epoch 00070: val_accuracy did not improve from 0.96304
Epoch 71/150
331/331 [==============================] - 33s 100ms/step - loss: 0.1144 -
accuracy: 0.9566 - val_loss: 0.1493 - val_accuracy: 0.9468

Epoch 00071: val_accuracy did not improve from 0.96304
Epoch 00071: early stopping
```

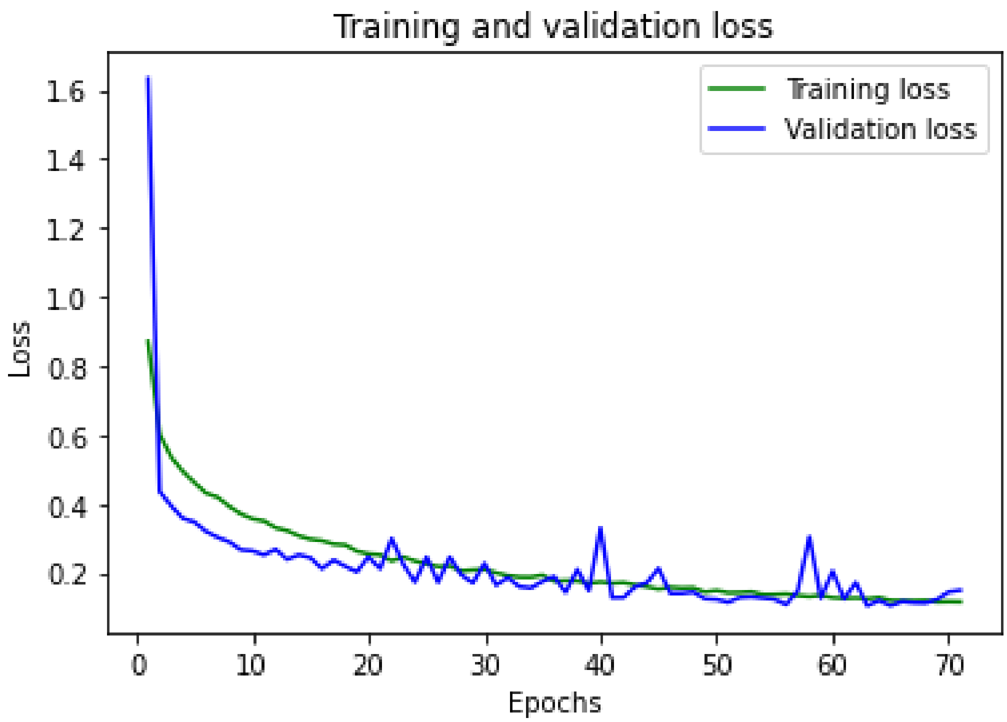
[ ]:
```
model.save(os.path.join(BASE_PATH, 'covid_classifier_result.h5'))
```

[ ]:
```
test_loss, test_acc = model.evaluate(test_generator)
print("Loss on test set: ", test_loss)
print("Accuracy on test set: ", test_acc)
```

```
72/72 [==============================] - 4s 51ms/step - loss: 0.1349 - accuracy:
0.9529
Loss on test set:  0.1348775327205658
Accuracy on test set:  0.9529463648796082
```

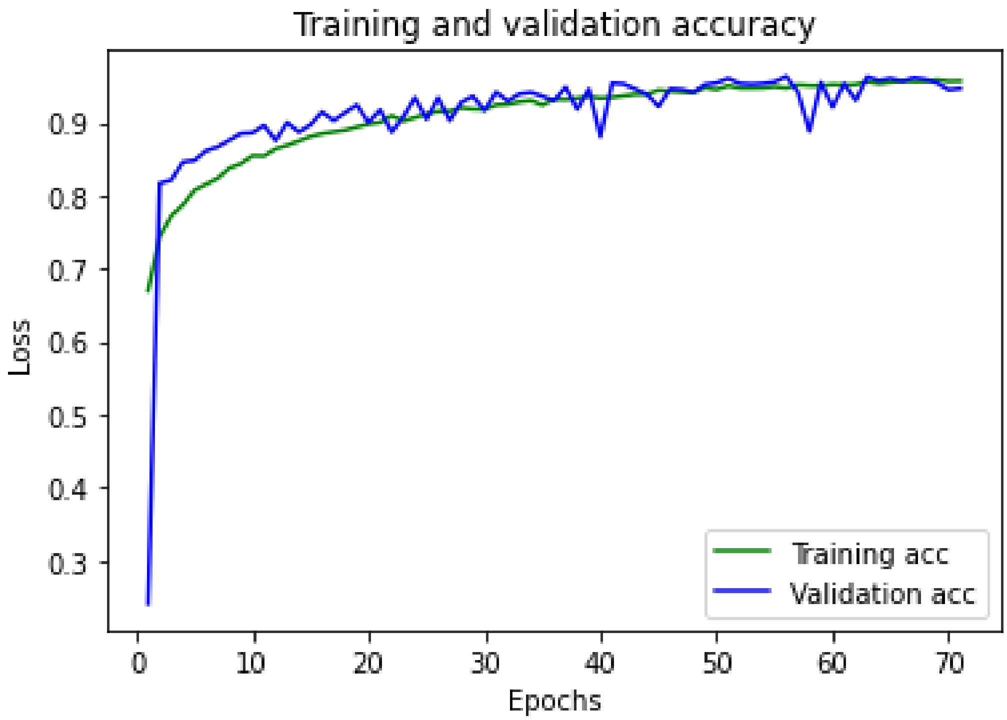
[ ]:
```
import matplotlib.pyplot as plt

acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(1, len(acc) + 1)
# bo is for blue dot.
plt.plot(epochs, loss, 'g', label='Training loss')
# b is for solid blue line
plt.plot(epochs, val_loss, 'b', label='Validation loss')
```

```
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

plt.show()
```



[ ]:

```
plt.clf()

plt.plot(epochs, acc, 'g', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

plt.show()
```



[ ]:
```
from sklearn.metrics import classification_report, confusion_matrix

X_test = []
Y_test = []
# Extract the data
for X, Y in test_generator:
    X_test.append(X)
    Y_test.append(Y)
X_test = np.array(X_test)
Y_test = np.array(Y_test)

predictions = model.predict_classes(X_test)
predictions = predictions.reshape(1, -1)[0]

print(classification_report(Y_test, predictions, target_names=[
      'Covid (Class 0)', 'Normal (Class 1)', 'Viral Pneumonia (Class 2)']))
```

[ ]:
```
import pandas as pd
import seaborn as sns

labels = ['covid', 'normal', 'viral_pneumonia']

cm = confusion_matrix(Y_test,predictions)
```

```
cm = pd.DataFrame(cm , index = ['0','1'] , columns = ['0','1'])
plt.figure(figsize = (10,10))
sns.heatmap(cm,cmap= "Reds", linecolor = 'black' , linewidth = 1 , annot =␣
↪True, fmt='',xticklabels = labels,yticklabels = labels)
```