**MODEL VGG16**

import os

import zipfile

import numpy as np

import tensorflow as tf

import matplotlib.pyplot as plt

from glob import glob

from tqdm import tqdm\_notebook

from keras.preprocessing.image import ImageDataGenerator

from keras.layers import Input, Lambda, Dense, Flatten

from keras.models import load\_model, Model

from keras.applications.vgg16 import VGG16

from keras.applications.vgg16 import preprocess\_input

%matplotlib inline

tf.\_\_version\_\_

from google.colab import drive

drive.mount('/content/drive')

# setup file structure

base\_dir = "/content/drive/MyDrive/chest\_xray/chest\_xray"

train\_dir = "/content/drive/MyDrive/chest\_xray/chest\_xray/train/"

test\_dir = "/content/drive/MyDrive/chest\_xray/chest\_xray/test/"

val\_dir = "/content/drive/MyDrive/chest\_xray/chest\_xray/val/"

print("Number of images in Train is {}".format(len(glob(train\_dir + "\*/\*"))))

print("Number of images in Test is {}".format(len(glob(test\_dir + "\*/\*"))))

print("Number of images in Validation is {}".format(len(glob(val\_dir + "\*/\*"))))

Number of images in Train is 5216

Number of images in Test is 624

Number of images in Validation is 16

IMG\_SHAPE = (224, 224,3)

base\_model = VGG16(input\_shape=IMG\_SHAPE, include\_top=False, weights='imagenet')

Downloading data from <https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5>

58892288/58889256 [==============================] - 1s 0us/step

model.compile(  loss='categorical\_crossentropy',

  optimizer='adam',

  metrics=['accuracy'])

train\_datagen = ImageDataGenerator(rescale = 1./255,

                                   shear\_range = 0.2,

                                   zoom\_range = 0.2,

                                   horizontal\_flip = True)

test\_datagen = ImageDataGenerator(rescale = 1./255)

training\_set = train\_datagen.flow\_from\_directory(train\_dir,

                                                 target\_size = (224, 224),

                                                 batch\_size = 32,

                                                 class\_mode = 'categorical')

test\_set = test\_datagen.flow\_from\_directory(test\_dir,

                                            target\_size = (224, 224),

                                            batch\_size = 32,

                                            class\_mode = 'categorical')

valid\_loss, valid\_accuracy = model.evaluate\_generator(test\_set)

plt.plot(history.history['loss'])

plt.plot(history.history['val\_loss'])

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

plt.plot(history.history['accuracy'])

plt.plot(history.history['val\_accuracy'])

plt.title('model accuracy')

plt.ylabel('accuracy')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

# summarize history for loss

plt.plot(history.history['loss'])

plt.plot(history.history['val\_loss'])

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

**RESNET50**

import os

import zipfile

import numpy as np

#import tensorflow as tf

import matplotlib.pyplot as plt

from glob import glob

from tqdm import tqdm\_notebook

from keras.preprocessing.image import ImageDataGenerator

from keras.layers import Input, Lambda, Dense, Flatten

from keras.models import load\_model, Model

from keras.applications.resnet50 import ResNet50

from keras.applications.resnet50 import preprocess\_input

%matplotlib inline

tf.\_\_version\_\_

base\_dir = "/content/drive/MyDrive/Kaggel/pneumonia\_data/chest\_xray"

train\_dir = os.path.join(base\_dir, "train/")

test\_dir = os.path.join(base\_dir, "test/")

val\_dir = os.path.join(base\_dir, "val/")

print("Number of images in Train is {}".format(len(glob(train\_dir + "\*/\*"))))

print("Number of images in Test is {}".format(len(glob(test\_dir + "\*/\*"))))

print("Number of images in Validation is {}".format(len(glob(val\_dir + "\*/\*"))))

Number of images in Train is 5216

Number of images in Test is 624

Number of images in Validation is 16

IMG\_SHAPE = (224, 224,3)

base\_model = ResNet50(input\_shape=IMG\_SHAPE, include\_top=False, weights='imagenet')

Downloading data from <https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet50_weights_tf_dim_ordering_tf_kernels_notop.h5>

94773248/94765736 [==============================] - 1s 0us/step

base\_model.trainable = False

base\_model.output

# useful for getting number of classes

folders = glob(train\_dir+'/\*')

# our layers - you can add more if you want

x = Flatten()(base\_model.output)

# x = Dense(1000, activation='relu')(x)

prediction = Dense(len(folders), activation='softmax')(x)

model = Model(inputs=base\_model.input, outputs=prediction)

model.summary()

model.compile(  loss='categorical\_crossentropy',

  optimizer='adam',

  metrics=['accuracy'])

train\_datagen = ImageDataGenerator(rescale = 1./255,

                                   shear\_range = 0.2,

                                   zoom\_range = 0.2,

                                   horizontal\_flip = True)

test\_datagen = ImageDataGenerator(rescale = 1./255)

training\_set = train\_datagen.flow\_from\_directory(train\_dir,

                                                 target\_size = (224, 224),

                                                 batch\_size = 32,

                                                 class\_mode = 'categorical')

test\_set = test\_datagen.flow\_from\_directory(test\_dir,

                                            target\_size = (224, 224),

                                            batch\_size = 32,

                                            class\_mode = 'categorical')

valid\_loss, valid\_accuracy = model.evaluate\_generator(test\_set)

print("Accuracy after transfer learning: {}".format(valid\_accuracy))

plt.plot(history.history['loss'])

plt.plot(history.history['val\_loss'])

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

plt.plot(history.history['accuracy'])

plt.plot(history.history['val\_accuracy'])

plt.title('model accuracy')

plt.ylabel('accuracy')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

# summarize history for loss

plt.plot(history.history['loss'])

plt.plot(history.history['val\_loss'])

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()