Using CNN and transfer learning to diagnose COVID-19 Pneumonia given chest X-ray image

• *Introduction: Why COVID-19 is wreaking havoc*

The COVID-19 has caused over 1 million confirmed cases and over 65 thousand deaths in the United States, and those numbers are still growing. Some cities, such as NewYork, are still severely affected by the pandemic. One of the most important reasons for the outbreak of COVID-19 is the lack of medical resources, such as the ventilator, health care workers, and doctors, who provide essential diagnosis. Although it is a little challenging to increase the number of ventilators or medical personnel with the techniques learned in CS540, it is possible to use CNN(Convolutional Neural Network) to help the treatment of COVID-19 and support the doctors.

• *How CNN can help with diagnosis of COVID-19*

The Center for Disease Control and Prevention(CDC) stated that COVID-19 is a respiration disease.[1] It often causes pneumonia, inflammation in the lung, and the patients' lungs would be filled with fluid and pus.[1] As a result, patients died due to suffocation caused by pneumonia. Therefore, the diagnosis of the presence of viral pneumonia is critical to the treatment of COVID-19.

Usually, doctors can tell if a patient gets pneumonia by checking if there are dark areas or shadows on the chest X-ray image. The figure1 shows the X-ray image of a healthy lung, and figure 2 shows the unhealthy lung. This process is vital for treatment, but is time-consuming at the same time. Moreover, the accuracy of diagnosis from doctor could decrease due to fatigue. CNN, which is good at finding the pattern in the image, can support doctors to tell if a patient gets pneumonia caused by COVID-19 in a short time with high accuracy.

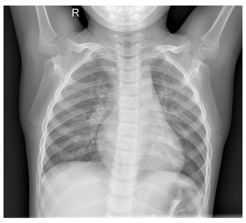


Figure 1

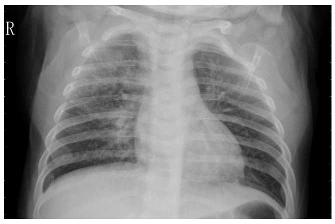


Figure 2

• *The stakeholders of the project*

Applying CNN in chest X-ray diagnosis can be beneficial for most of the doctors and patients who are suffering from COVID-19. Some doctors committed suicide due to the high pressure and large workload caused by COVID-19, and many patients cannot obtain treatment because of the lack of a doctor's diagnosis.

If CNN can help doctors make accurate and fast diagnoses based on the chest X-ray images, the workload of doctors will reduce, and more patients can be treated since CNN speeds up the diagnostic process.

• *General plan*

To apply CNN, I need to obtain the X-ray images of both the healthy lung and the lung with viral pneumonia. After getting the data, I'll need to import the appropriate pre-trained convolutional network model with the corresponding weights and add a few fully connected dense layers by myself. Then, I plan to train the model with my data images and labels. I also plan to separate data into three sets: training set, tuning set, and test set. This allows me to evaluate the performance of the model and make improvements.

• *The main technical steps and the verifiable milestones of those steps*

Step 1: Search and get the chest X-ray images of lungs with corresponding labels. Obtaining enough X-ray images helps train and test the model built.

Step 1 evaluation: At the end of step 1, the data file should contain 50 X-ray images of healthy lungs and 50 X-ray images of lungs with viral pneumonia.

Step 1 time estimation: 3 days

Step 2: Set the labels for images. If the image shows a healthy lung, set the corresponding label to be 0. If the image shows an unhealthy lung, set the corresponding label to be 1. Then, randomly separate the images into three sets(training, tuning, testing) with their labels. The training set, tuning set, testing set should contain 60, 20, 20 images, respectively.

Step 2 evaluation: At the end of step 2, I should have 3 data sets. The images training set, tuning set, testing set should be loaded as (60,224,224,3), (20,224,224,3), (20,224,224,3) respectively. The labels should be 1d arrays that contain only 1s and 0s to represent if the lung in the corresponding image is healthy.

Step 2 time estimation: 3 days

Step 3: Do some research about the pre-trained convolutional model. Find and import the suitable pre-trained model from keras.application.

Step 3 evaluation: At the end of step 3, I should have a suitable pre-trained convolutional base model downloaded and imported, so that I can add a few fully connected layers later in step 4. This is the key to transfer learning. Can use model.summary() to check if the layers are correctly added.

Step 3 time estimation: 3 days

Step 4: Add several fully connected layers including a Flatten layer, a Dense layer with 64 nodes and 'relu' as activation function, and a Dense layer with 10 nodes and 'softmax' as activation Function so that the final output will be a discrete probability distribution over the two target classes(healthy/unhealthy).

Step 4 evaluation: At the end of step 4, I build a model that is the combination of the pre-trained model and fully connected layers. Can use model.summary() to check if the layers are correctly added.

Step 4 time estimation: 3 days

Step 5: Train the model with the training set and tuning set.

Step 5 evaluation: At the end of step 5, I should have a trained model that can accurately identify the images in the training set and tuning set.

Step 5 time estimation: 3 days

Step 6: Use the model to predict the images in the test set and record the success rate(number of correct prediction/20). If the success rate is low, I can try different fully connected layers. For example, I may try to add Dense layers with a different number of nodes or different activation function.

Step 6 evaluation: At the end of step 6, I should have a model that is accurate(about 90% of accuracy). I'll save the model and share it online.

Step 6 time estimation: 2 weeks. I think testing and improving the model to achieve 90% of accuracy is time-consuming.

• *Preliminary results*

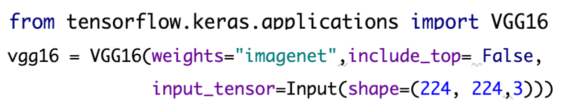
I currently have done most of the research work and a few coding for step1(data collecting), step 3(download and use the pre-trained model) and step 4(add fully connected layers).

Preliminary results of step 1: I searched online and find a shared image set that contains over 2000 chest X-ray images of both healthy lung and lung with pneumonia caused by COVID-19 at kaggle.[2] I download 50 images for both health and unhealthy lung.

Preliminary results of step 3: I have searched the keras application library and find out that VGG16[3] is a suitable pre-trained model for my project. Before importing VGG, updated the python certificate first to prevent URL fetch error. The code is shown in figure 3 and figure 4 is the layer structure of VGG 16.

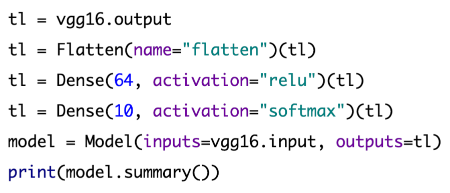
Preliminary results of step 4: I added the Flatten layer and Dense layer the way similar to project 9 and 10. The difference is this time it's about transfer learning. The structure of the model after adding fully connected lays is shown in figure 6 and the code is shown in figure 5.

Since I have partially built the model and I have obtained the data(images) for training, tuning, testing. I think my project is plausible.



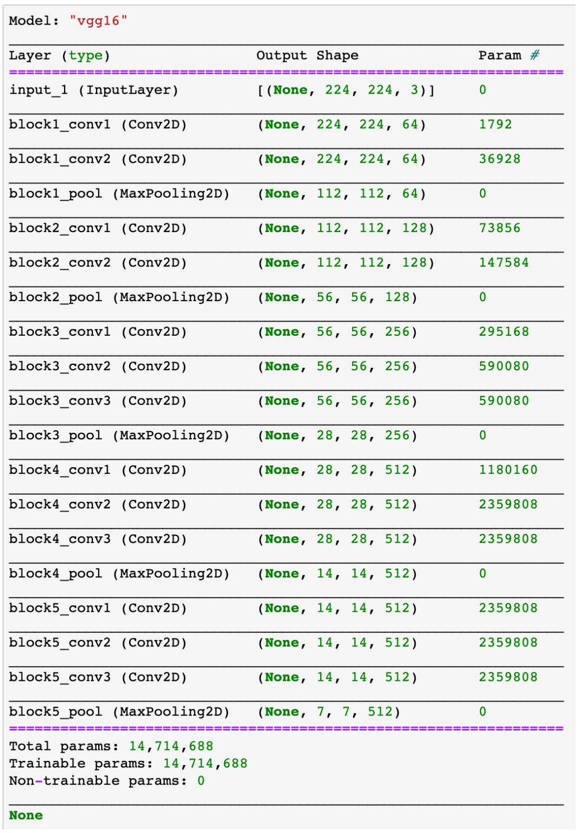
```
from tensorflow.keras.applications import VGG16
vgg16 = VGG16(weights="imagenet",include_top= False,
              input_tensor=Input(shape=(224, 224,3)))
```

Figure 3



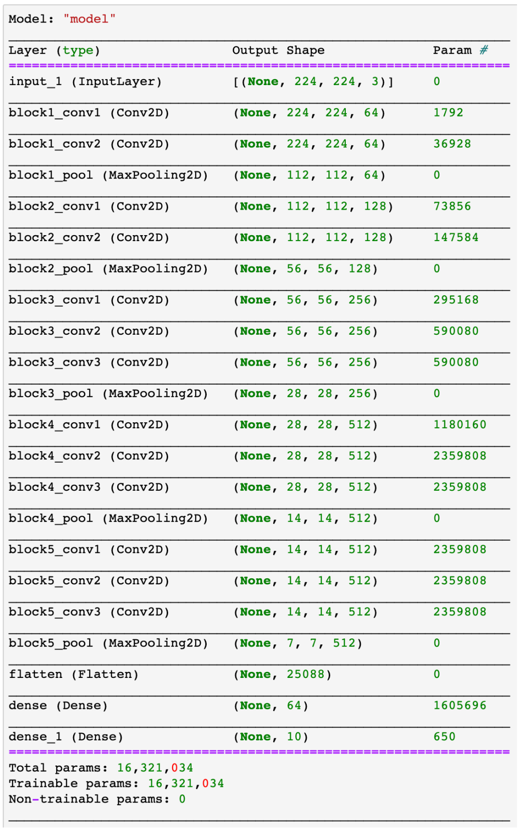
```
tl = vgg16.output
tl = Flatten(name="flatten")(tl)
tl = Dense(64, activation="relu")(tl)
tl = Dense(10, activation="softmax")(tl)
model = Model(inputs=vgg16.input, outputs=tl)
print(model.summary())
```

Figure 5



```
Model: "vgg16"
_________________________________________________________________
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_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_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_pool (MaxPooling2D)   (None, 7, 7, 512)         0
=================================================================
Total params: 14,714,688
Trainable params: 14,714,688
Non-trainable params: 0
_________________________________________________________________
None
```

Figure 4



```
Model: "model"
_________________________________________________________________
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_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_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_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
dense (Dense)                (None, 64)                1605696
_________________________________________________________________
dense_1 (Dense)              (None, 10)                650
=================================================================
Total params: 16,321,034
Trainable params: 16,321,034
Non-trainable params: 0
_________________________________________________________________
```

Figure 6

• *Similar projects*

It turns out that many data scientists are using CNN to analyze the X-ray image and provide diagnosis. A similar project also uses VGG16 as pre-trained model.[4] The difference between my project and that one is the number of data sets used. The author of that project separates the data images into 2 set: training set and a test set to train the model, while I separate the data images into 3 sets: a training set, a tuning set, and a test set. One research team also tried to analyze chest X-ray image with CNN, but they used VGG19 as the pre-trained model.[5] Some other research teams used several different pre-trained model such as MobileNet v2, Inception, Xception, and Inception ResNet v2.[6] One research team even apply CNN on CT images and still obtain a high accuracy of diagnosing COVID-19.[7]

Reference

[1] Villines, Zawn. "Pneumonia and COVID-19: Relationship, Risks, and More." *Medical News Today*, MediLexicon International, www.medicalnewstoday.com/articles/pneumonia-and-covid-19.

[2] https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia by Paul Mooney

[3] K. Simonyan, A. ZissermanarVery, "Deep Convolutional Networks for Large-Scale Image Recognition" Xiv:1409.1556

[4] https://www.pyimagesearch.com/2020/03/16/detecting-covid-19-in-x-ray-images-with-keras-tensorflow-and-deep-learning/ by Adrian Rosebrock

[5] Md. Rezaul Karim .et al, "DeepCOVIDExplainer: Explainable COVID-19 Predictions Based on Chest X-ray Images" URL:https://arxiv.org/pdf/2004.04582.pdf

[6] Ioannis D. Apostolopoulos & Tzani A. Mpesiana, "Covid-19: automatic detection from X-ray images utilizing transfer learning with convolutional neural networks",URL:https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7118364/

[7] Halgurd S. Maghdid .et al, "Diagnosing COVID-19 Pneumonia from X-Ray and CT Images using Deep Learning and Transfer Learning Algorithms", URL: https://arxiv.org/abs/2004.00038