Recognizing COVID-19 from Lung CT Scan Images by various base models

Botond Barta
Budapest University of Technology and Economics
b.botond25@gmail.com
CSGJK1

Lilla Matyasi
Budapest University of Technology and Economics
lilla.matyasi@gmail.com
X986BW

Dorina Lakatos
Budapest University of Technology and Economics
dorinapetra@gmail.com
FWH5YD

***Abstract*—The Coronavirus disease (COVID-19) is a quite fast-spreading virus that caused many serious problems in the year 2020 and unfortunately this pandemic has not ended. The reverse transcription-polymerase chain reaction (RT-PCR) tests do not always show the correct prediction. In this paper we tried to create a model that can predict if a patient is infected or not from a lung CT scan. We trained 4 models each based on a pretrained network to see which one can give the best results. The best model is based on the VGG16 pretrained network with a high 99% accuracy on the test set.**

I. Introduction

As this year (2020) was all about the coronavirus we thought that this topic will be an appropriate choice for this year. Right now (11.12.2020) the coronavirus has infected more than 71 million people and unfortunately caused the death of approximately 1.6 million people all over the world, therefore this is a serious global crisis at the moment and will be in the next few years (or maybe months).

The application of Deep Learning in this case seems obvious, because once if the networks has been trained well with the datasetset, the model can decide in seconds if a patient is infected by the coronavirus or not with a very high confidence. Using our model can be really helpful for the doctors especially in times like this when they are really overwhelmed.

II. Background and motivation

We chose this project because we had a personal motivation: one of our teammates' mothers had a lung Computed Tomography (CT) in September and the doctors could not really decide if it was covid-19 or not. She had many symptoms; however, all of her tests came back negative. One doctor said it was Covid-19, and she had to go to the ICU immediately, however another said that it was nothing.

Therefore, we decided to make a model which can decide with a quiet good accuracy whether the patient has Covid-19 infection or not. Figure 5 shows 2 CT scans from which we tried to predict if it was Covid-19 or not.

III. Data

We are using data from medRxiv provided by Mohammad Rahimzadeh. The dataset is originally 20 GB, but it contains a lot of images which are not very useful for us as they were captured in the beginning or at the end and the inside of the lungs are not observable. Unfortunately, we were not able to use the whole dataset as we did not have the appropriate resources for this project, we explain this in detail at the Limitations section.

They separated the dataset into 5 folders for training validating and testing. We chose the first fold, although we did not use the same rates, they provided us because the validation set was the subset of the test set and we did not agree with that. We separated the images in a unique way shown in Figure 7. We used different rates for separating the Covid and Normal images due to the different sizes of the Covid and Normal dataset (the number of Normal scans is much larger than of the Covid ones).

The Covid CT dataset contains 9776 healthy and 2282 Covid CT scans from 95 COVID-19 patients and 282 healthy people. The original images are 16-bit uint grayscale with 512*512 pixels resolution. Figure 3 and Figure 4 shows a lung CT scan of a Covid-19 infected patient and a healthy person.

IV. Pretrained network

A. *ResNet50V2*

ResNet is one of the most powerful deep neural networks which has achieved outstanding performance results in the ILSVRC 2015 classification challenge. It is a neural network that builds on pyramidal cells. The ResNet50 uses bottleneck design. ResNet50V2 is all about using the pre-activation of weight layers instead of post-activation.

B. *DenseNet169*

The DenseNet169 model is one of the DenseNet group of models designed to perform image classification. This model has several advantages: they alleviate the vanishing-gradient, problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.

C. *VGG16*

VGG16 is a convolutional network for classification and detection. The input to cov1 layer is of fixed size 224 x 224 RGB image. The images are passed through some convolutional layers with a very small receptive field. Finally, some fully connected layers follow the convolutional ones with ReLU activations.

D. *InceptionV3*

Inception v3 is a widely used image recognition model that has been shown to attain greater than 78.1% accuracy on the ImageNet dataset. The model is made from symmetric and asymmetric building blocks (convolutions, average pooling, max pooling, dropouts, concats, fully connected layers)

V. Our Model

At first, we tried to build our own model based on general templates, that had some 2D convolutional layers as the input with Maxpooling between them, after that, we wanted to flatten the data and use dense layers as the output. However, we could not reach 85% with our model any the less we tried hyperparameter optimization. Therefore, we decided to try with pretrained models.

After that, we tried 4 different pretrained models (ResNet50V2, DenseNet169, VGG16 and InceptionV3) to find out which gives the best results.

After the trained model we put 2 Dense layers each followed by a DropOut layer for regularization purposes. The last layer is also a Dense layer too which is the output using sigmoid activation. The other Dense layers are using ReLU activations and their number of neurons are between 16 and 128 and the parameter of the DropOut layer is between 0.1 and 0.4. The structure of our model is shown in Figure 6.

We used the hyperparameter optimization to find the best model. For this we used the HyperBand from Keras.

As the dataset has only 2 classes (COVID or Normal) we decided to use binary classification so that our model last Dense layer has only one neuron with sigmoid activation.

We used transfer learning, we set the base model's layers trainability to false, and hyperparameter optimized the new dense layers with Nadam optimizer. Afterwards we saved the model that gave the best validation accuracy. Then we fine-tuned the whole model for some more epochs with SGD optimizer and early stopping.

The ResNet50V2 somehow did not work out as we thought it would no matter how hard we tried. So, we decided to handle it another way: the base model's layers were trainable the whole time, this was the only way to achieve higher accuracy on the validation data.

VI. Experimental results

The parameters and the best results are detailed in Figure 1. We show in the table the number of neurons in the 2 Dense layers, the parameters of the Dropout layers, the loss and accuracy for both the validation and test set and the false negative rate which is a very important factor among the results. The test accuracies and losses are plotted in a diagram too in Figure 8 and Figure 9.

We plotted the confusion matrices of all trained models to find out which has the smallest false negative rate. The matrices are shown in Figure 2.

Overall, we decided to use the VGG16 as the base model because this one had the best results on the validation and test dataset. We focused on the false negative rate as in our opinion it is more crucial to recognize positive results (even if they are false positive) than to miss the infected patients which can cause that the patient will not be cautious and will not get the appropriate treatments.

The best model has predicted 0 patients as false negative out of 391 which we believe a quite good even if it is not exactly shows the real value, but we hope our test accuracy is close to the real one.

Our best model achieved an outstanding 99.01% accuracy on our test dataset and 99.22% on the validation dataset with a low false negative rate.

VII. Limitations

Unfortunately, we did not have a GPU in our computers, so we had to use Google Colab which has its own limitations. For example, after the maximum of 12 hours our session will be disconnected, and all of our trained networks will be gone. Especially the hyperparameter optimalization took lots of time, therefor it was hard to apply it to all the four models in time. The disk of the Google Colab is only 68.4 GB and we had to store a lot of data as we used keras-tuner's hyperparameter optimization and often we ran out of space, so we had to pay attention to this also because we did not want to lose everything that were trained till then.

Optimizing with InceptionV3 as the base was the slowest one, as it is one of the biggest model of all the four we used. That is why the optimizing parameters are limited at this one: Dropout rate from 0.2 to 0.3 by step 0.1 and number of neurons in the fully connected layers from 48 to 128 and 16 to 128 by step 32.

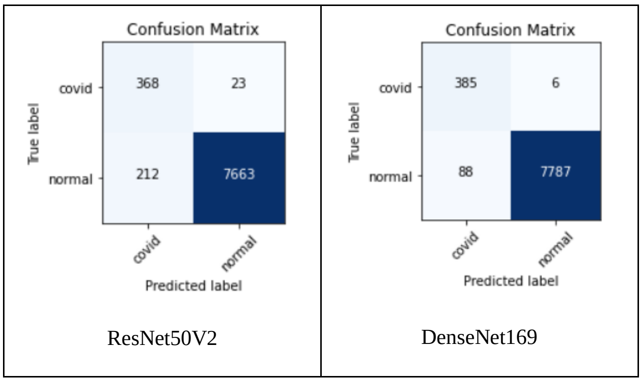
VIII. Conclusion

In conclusion we really think our model can decide with a rather high accuracy whether a patient has Covid-19 infection or not, from a single chest CT. We believe that our model could be used in hospitals effectively to evaluate a CT automatically. Another thing we learnt is to always try pretrained models as the base, because they have learned patterns for the images.

Interestingly our models did not give the same predictions for our 2 test images. While the network that used VGG16 or InceptionV3 was very sure (~96%-99.9%) about that our patient was healthy, the model that based on DenseNet169 was quite confident (~65.3%-93.4%) that our patient was infected by Covid-19. The model that used Resnet50V2 was quite unsure about the prediction but predicted no Covid-19 infection. Overall, this explains why the doctors could not agree with each other. Even well-trained models with very high accuracies could not agree in this issue. As we chose the model with VGG16 we believe that our teammates' mother was healthy when the CT scans were made.

	ResNet50 V2	***DenseNet***	***VGG16***	***Inception V3***
dense_1_units	112	96	80	112
dropout_1	0.2	0.2	0.2	0.2
dense_2_units	16	96	112	80
dropout_2	0.3	0.2	0.1	0.2
val_accuracy	0.9874	0.9912	0.9922	0.9570
val_loss	0.0321	0.0198	0.0183	0.1401
test_accuracy	0.9716	0.9886	0.9901	0.9674
test_loss	0.0923	0.0365	0.0260	0.099
false negative rate	0.0588	0.0153	0	0,0511
13TUDO.tif	0.7505	0.347	0.9971	0.9919
5TUDO.tif	0.5888	0.0655	0.9993	0.9599

Fig. 1. The parameters and the best results of the trained models. We also added the predictions of th 2 CT scans we wanted to test from the begginning. Interesting that the models gave really different results. The formulas are shown in Figure 11.



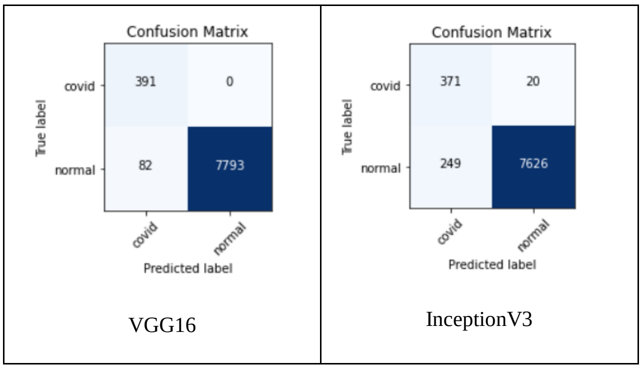


Fig. 2. Confusion matrices of all the trained models. We focusoed on the number of false negative cases because it is better to diagnose a patient as positive when they are healthy than diagnose them negative when in reality they are positive and will not be careful or will not get the necessary treatments. Moreover they can spread the virus without their knowledge.

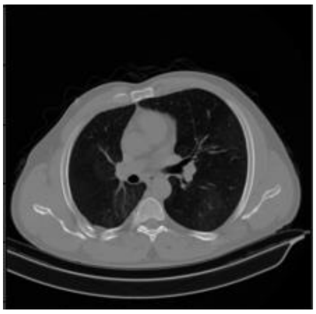


Fig. 3. The CT scan of a COVID-19 patient from the dataset we used for training

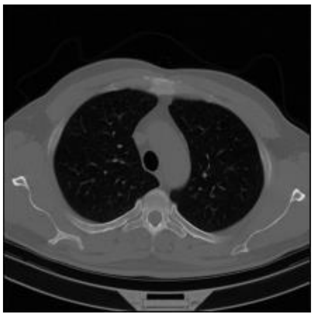


Fig. 4. The CT scan of a healthy patient from the dataset we used for training

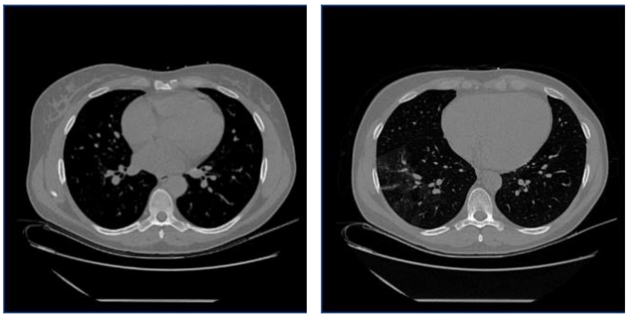


Fig. 5. The two type of pictures we want to test with our model. These CT scans are one of our teamates' mother's. We chose this two image because they are quite different from each other so we can predict more real diagnostics.

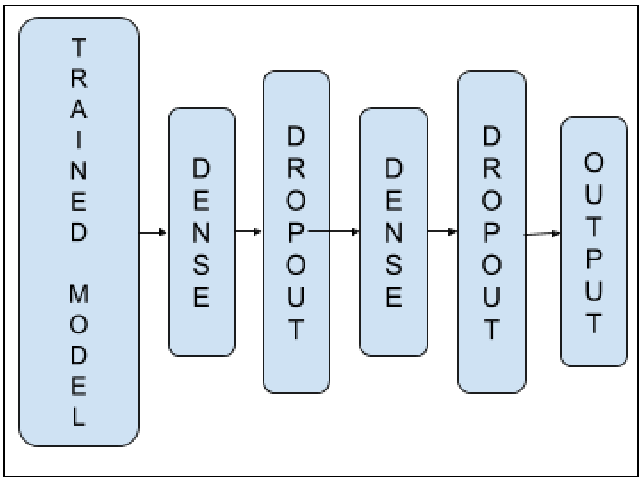


Fig. 6. The general structure of our models. We put 2 Dense layers after the trained base model, each followed by a Dropout layer. The ouput is a Dense layer with one neuron as we used binary classification.

	Train	Validation	Test
Covid	1500 (~65%)	391 (~17,5%)	391 (~17.5%)
Normal	1500 (~15%)	391 (~4%)	7875 (~80.6%)

Fig. 7. The rates we used for separating tha data. We did not use the same rates for Covid and Normal images because the dataset contains a lot more Normal than Covid images.

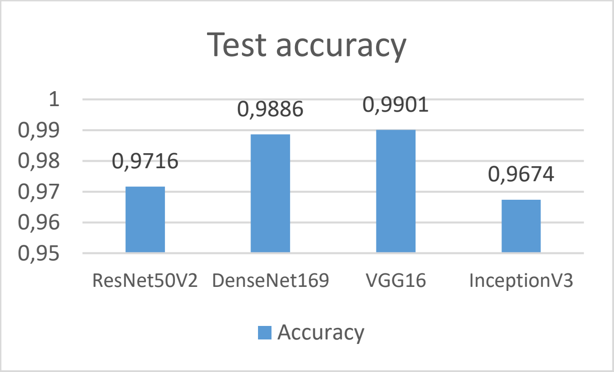


Fig. 8. The test accuracies of the trained models.

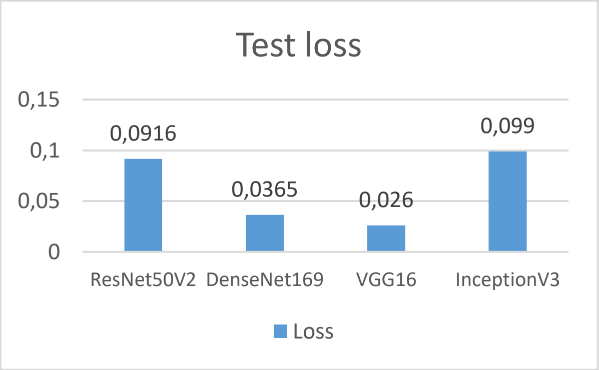


Fig. 9. The test losses of the trained models.

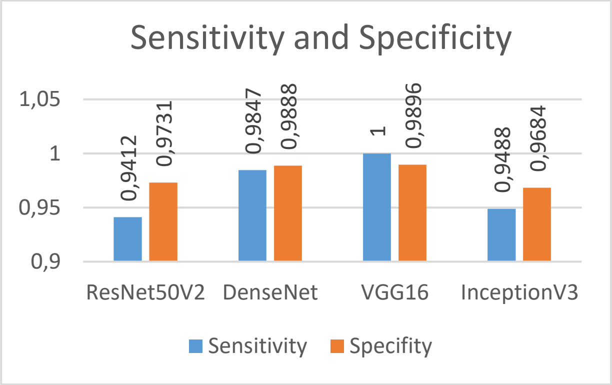


Fig. 10. The sensitivity and specificity of the trained models. We used the formulas shown in Figure 11.

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$False\ Negative\ Rate = \frac{FN}{FN + TP}$$

Fig. 11. The formulas we used to describe the results of our models.

IX. References

He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (2015-12-10). "Deep Residual Learning for Image Recognition". arXiv:1512.03385

Rahimzadeh, M.; Attar, A.; Sakhaei, M. A Fully Automated Deep Learning-based Network For Detecting COVID-19 from A New And Large Lung CT Scan Dataset. Preprints 2020, 2020060031 (doi: 10.20944/preprints202006.0031.v3).

Sohrabi, C. et al. World Health Organization declares global emergency: a review of the 2019 novel coronavirus (COVID-19). Int J. Surg. 76, 71–76 (2020).

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385,2015.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556,2014.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,pages 1–9,2015.

Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger. Densely Connected Convolutional Networks. arXiv:1608.06993.

Karen Simonyan, Andrew Zisserman. Very deep convolutional networks for large-scale image recognition arXiv:1409.1556

Abien Fred Agarap. Deep Learning using Rectified Linear Units (ReLU) arXiv:1803.08375

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res., 15(1):1929–1958, January 2014