UNIVERSITY OF PIRAEUS
Department of Digital Systems

Covid-19 Detection from X-ray Images

Georgios Panagiotakopoulos MSC
Information Systems and Services,
University of Piraeus,Piraeus,
Greece, georgepanlos@gmail.com
Registration Number: me2030

Abstract

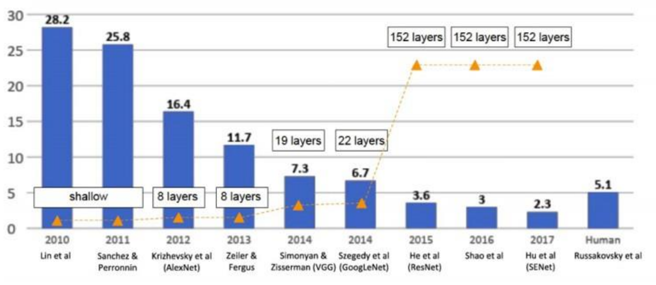
The Covid-19 pandemic is one of the most important (if not the most important) health threats today. In the last year and a half our lives have undergone great changes due to the increased protection measures and the restriction of movements / lockdowns. Its easy transmissibility and the risk to vulnerable groups have contributed to this situation. However, despite increased protection measures, more than 180 million cases have been reported worldwide to date, of which more than 4 million have been killed. Five vaccines (which are available to the population) have been developed to control the virus, as well as various tests for its diagnosis, of which the RT-PCR molecular tests are more sensitive and accurate, they can detect the virus in asymptomatic persons. The aim of this study is to design and create a system for automated diagnosis of the disease by categorizing lung X-rays using Convolutional Deep Neural Networks, as well as the creation of software / application for mobile phones which utilizes the above diagnostic system by providing him with X-ray photos in order to make a diagnosis.

1) Introduction

The Coronavirus, Covid-19 [1] was first reported in Wuhan, China in late December 2019 and is a respiratory disease caused by the new SARS-CoV-2 coronavirus (severe acute respiratory syndrome Coronavirus 2 (SARS-Cov-2) 2019 [4]. The name Covid-19 is the short name given to it by the World Health Organization. The original origin of the disease seems to be from an animal organism however the exact origin has not been confirmed. Symptoms of this disease are high fever (above 38 °C) dry cough, difficulty breathing, fatigue, body pains, sore throat and diarrhea for some people. Sudden loss of smell, without nasal obstruction and complete disappearance of taste, are also symptoms observed in patients. In people who develop more severe forms, respiratory difficulties are identified, which can lead to hospitalization in intensive care and death. The characteristics of Covid-19 that make it so dangerous are the way it is transmitted and that the onset of symptoms can last up to 15 days. It is also worth noting that there are cases of patients who are asymptomatic with the result that they are carriers of the virus that are aware of it. The coronavirus is transmitted through droplets secreted during speech, a cough or sneezing. To combat the spread of the virus, the World Health Organization advises taking certain precautions such as frequent hand washing with soap or hydroalcoholic solution, avoiding close contact, such as kissing or shaking hands, with people coughing or sneezing and sneezing. one use. In order to combat the spread of Covid19, some countries restricted movement and activity in cities, while others adopted more stringent measures such as lockdown until a sufficient proportion of the population was vaccinated. At the time of writing, 5 vaccines have been developed [2] (such as Pfizer-BioNTech Moderna and others) which are available for administration. The most effective method for diagnosing the virus is RT-PCR molecular tests [3] that detect the genetic material, RNA, of the virus. They have greater sensitivity and accuracy, they can detect the virus even in asymptomatic individuals, but they are quite expensive and time consuming to produce results. The scientific community has also developed two more economical and direct tests (Rapid and Self Test.) [5] which, however, do not have very high virus detection rates. To combat the above problems medical image processing was used to confirm positive patients in Covid19. Using chest X-rays and Compute Tomography scans provided as input to Computer Vision Systems and models have been developed that can assist radiologists in diagnosing the disease. The main purpose of this study is to establish a disease diagnosis system using deep learning techniques in X-rays in order to diagnose a patient with Covid19. Also the second part of the study aims to develop a Mobile Application [8] which utilizes the above model in order to scan images and make diagnoses. For the design of the diagnostic mechanism i created a Convolutional Neural Network which was compared to pre-trained neural networks (like MobileNetV2, VGG16, DenseNet169, InceptionResNetV2, DenseNet201, InceptionV3, VGG19 and ResNet50) which were configured using the transfer learning method in order to optimize their performance in our problem. For the training of the above models, the data set "COVID-19 Radiography Dataset" was used, which consists of images of lung X-rays which have been divided into 4 categories. The mobile application was actualized using the CoreML framework provided by Apple and supported by its devices in order to load trained neural networks on the device and use them through the application to make predictions. The application loaded the 3 models that presented the best forecast performance and were tested for their efficiency and for the resources consumed by the device. The paper is followed by an extensive bibliographic review, an explanation of the methods and architectures used to train the neural networks as well as a detailed presentation of the mobile application created.

2) Authors and Affiliations

The use of machine learning to diagnose diseases from medical images is a field that has been developing for several decades. On the other hand, the continuous evolution of neural networks and in particular their variants such as Deep Neural Networks (DNN) [6] as well as Convolutional Neural Networks (CNN), make them one of the leading technologies in image recognition and classification [10]. A very important pre-trained model that has contributed a lot to the improvement of neural networks for image categorization is ImageNet [11] during which the first convolutional deep learning neural network was proposed by Alex Krizhevsky, also known as AlexNet [13]. This specific model managed to achieve an accuracy of 84.7% in the competition's dataset against 73.8% which was the most prevalent implementation so far. Since then, several architectures have been created that follow the logic of deep Convolutional networks and improved the performance of the competition such as VGG [13] and ResNet [14] which we will use in our research.

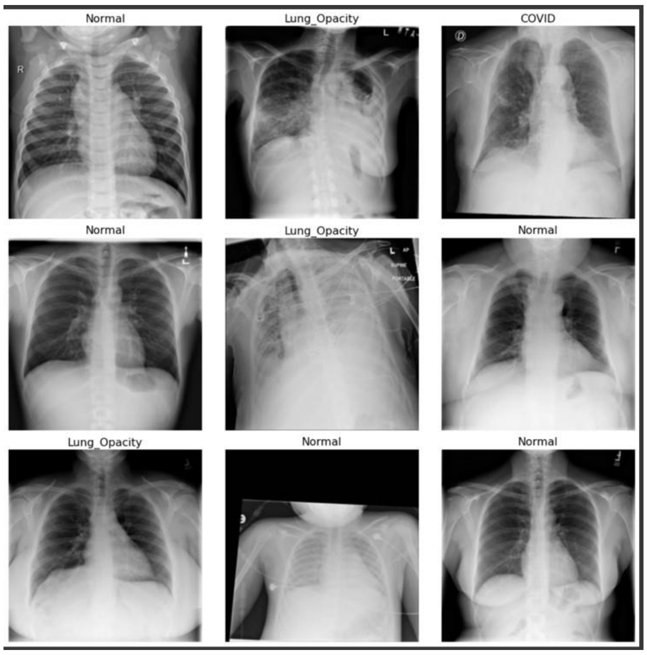


Architectural results proposed for ImageNet

Combining those two fields above, several scientific studies have been proposed for the diagnosis of diseases using CNN such as for the diagnosis of breast cancer [15] and for the diagnosis of heart disease [16]. In the concerning diagnosis of Covid-19 field, several architectures have already been proposed which mainly use transfer learning [17] for the training of their neural networks. An automated method proposed by Narin et al [18] where 50 X-rays were measured in 224x224 dimensions, into three DCNN (Deep Convolutional Neural Network) architectures. In the aforementioned architectures, the best results were observed by the ResNet50 architecture (with 98% accuracy). Another important published study has been presented by J Biomol Struct Dyn, in which transfer learning was used to classify 6087 images. Those images were divided into 3 categories (2780 bacterial pneumonia, 1724 coronavirus, and 1583 normal). Various pre-trained machine learning models were used for image classifications with better results showing Inception_Resnet_V2 [19] (92% accuracy).

3) Data Description

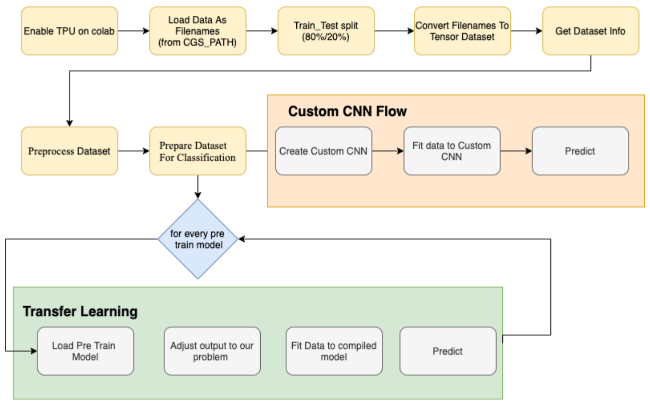
The "COVID-19 Radiography Dataset" dataset [20], which developed by a team of researchers from Qatar universities in collaboration with specialist doctors was used to train the models. This dataset consists of 21,165 X-rays which have been divided into 4 categories. In detail, the dataset contains 2886 X-rays labeled Covid-19, 4839 X-rays labeled Lung Opacity, 1096 X-rays labeled Viral Pneumonia and 8111 X-rays labeled Normal. Our concern is to distinguish X-rays of patients with Covid-19 from those with other lung diseases or who are healthy.



Sample X-rays contained in the dataset together with the corresponding label

4) System Architecture

For the implementation of the present study, a system was designed for the efficient reading and preprocessing of the data so that they come in a suitable form to be given as input to the algorithms. More detail:


Enable TPU on colab
Load Data As Filenames (from CGS_PATH)
Train_Test split (80%/20%)
Convert Filenames To Tensor Dataset
Get Dataset Info
Preprocess Dataset
Prepare Dataset For Classification
Custom CNN Flow
Create Custom CNN
Fit data to Custom CNN
Predict
for every pre train model
Transfer Learning
Load Pre Train Model
Adjust output to our problem
Fit Data to compiled model
Predict

System Architecture Diagram

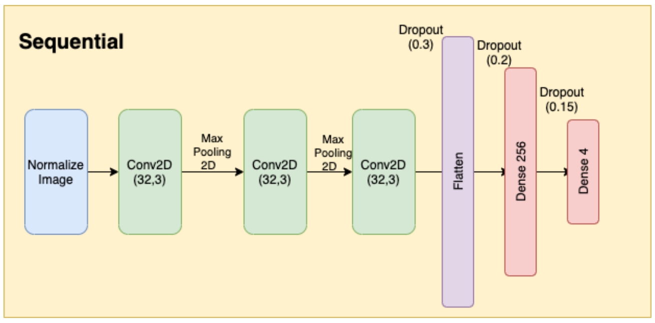
1. Algorithms were originally developed and executed using the Colab tool [21] which is an online platform on which we can write and execute code in the form of a Jupyter Notebook which can be easily shared between student and team teacher.
2. In order to speed up the training phase, we ran our algorithm on TPUs (Tensor Process Units) [22] which are provided free of charge by google via Colab. TPUs are specially designed computing units that are used to speed up linear algebra calculations that are responsible for the greater workload in Convolutional Neural Network applications as well as general machine learning problems.
3. In order for TPUs to speed up data reading, they must be stored in Google Cloud Storage (CGS). Kaggle Datasets are available on Google Cloud Storage, running the notebook via the Kaggle Dataset we want to study. So, we load the filenames (paths) from Kaggle CGS.
4. Separation of data into Train and Test sets with proportions of 80% for the train set and 20% for the test set. We separate the filenames into train and test sets with the train_test_split method of the sklearn Library.
5. We Convert train_filenames and test_filenames (which at this stage are path arrays) to Tensor datasets. We do this in order to optimize the transformation process that will follow.

6. We display some characteristics of the dataset such as the total samples for train and for test set, as well as how many samples there are per category (Covid, Normal, etc.) in each set of train and test set.
7. Pre-processing data in order to get it in a suitable format to be used by algorithms: First we export the label for each image through the filename path. (For example: dataset_path / Lung_Opacity / Lung_Opacity-233.png). Then we load the image from filepath, after,we decode it, then, we convert it to float32 and resize it to dimensions [299x299] that we will use as input to our algorithms.
8. Preparing for Classification. First of all, we use caching in our data because we know we have a small dataset, then, we shuffle the train dataset to avoid overfitting and we ensure that the data generator can provide at least $\boldsymbol{steps\_per\_epoch} * \boldsymbol{epoch}$ data using the repeat () method, we determine the batch size given by the relation: $\boldsymbol{batch\_size} = \mathbf{16} * \boldsymbol{num\_of\_tpu\_cores}$. Finally, we activate the prefetch next batch function which loads the next batch while the model is being trained using the autotune prefetch buffer size.

All steps above, were implemented in common for all the neural networks we trained, then we will describe in detail the stages of training and prediction using neural networks.

5) Custom CNN

In order to make a prediction / diagnosis of the X-ray classification, an attempt was made to construct an improvised Convolutional Neural Network that follows the following architecture:

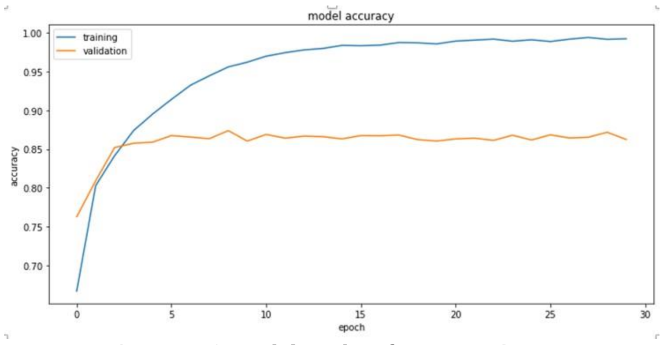


Improvised architecture of a CNN

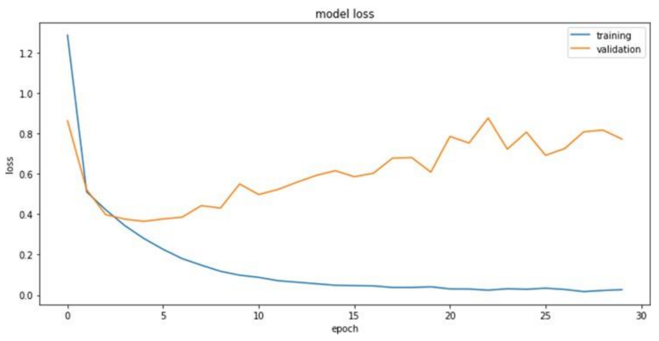
1. First of all, our neural network consists of a batch normalization layer which normalizes its input so that the output of this layer has an average value close to 0 and a standard deviation of 1.
2. Then the first Convolutional layer is applied, as it accepts a two-dimensional table to which it applies the act of convolution using 32 filters of 3x3 dimension.
3. Then, we apply max pooling to the result of the previous convolution. What max pooling practically does is apply a filter over the image and maintain the maximum value displayed on that filter. This method is used to compress the effect of convolution.
4. Then, we apply the 2nd convolutional layer which accepts the result that has emerged after the dropout and applies the act of convolution with 64 different filters of 3x3 dimensions.
5. We repeat the max pooling implementation for the result of the previous convolution.
6. We apply the last convolutional level in our network which accepts the result that has emerged after the dropout and applies the act of convolution with 128 different filters of 3x3 dimensions.
7. We repeat the max pooling implementation for the result of the previous convolution.
8. Then, we flatten the result obtained from the convolution by applying the dropout method with probability of 0.3. The dropout method "Deactivates" one neuron in each repetition with a probability that we define in other to force the other neurons to do their "execution". Also, it gives weight to the inputs of the other neurons. This procedure is applied to avoid overfitting.
9. Finally, we apply 2 fully connected dense layers of which the first has 256 neurons and accepts as input the result of flatten after a dropout has been applied with a probability of 0.2 and advances its result to the second which has 4 neurons (as many as the class we want to predict) with dropout 0.15.

This neural network was trained in the train set in which we set for 30 epochs and was used to predict the test set having the following results:

We observe in the accuracy / epoch diagram below, that after 10 epochs, our neural network has reached the maximum accuracy it can achieve and now all the values are moving around it.



Accuracy / Epoch line chart for custom CNN



Loss / Epoch line chart for custom CNN

In In contrast to the loss / epoch line chart, we observe that from epoch 10 onwards the loss increases significantly, which indicates overfitting. So, we conclude that in order to have a more efficient model, an ideal number of epochs would be 10.

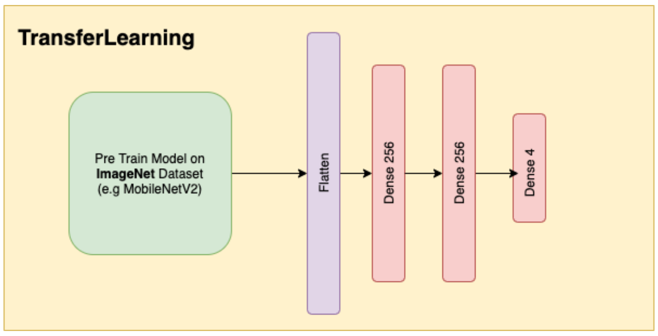
Model	Val Accuracy	Val Loss	Total Epoch
Custom Cnn	0.8603	0.5499	10
Custom Cnn	0.8625	0.7722	30

Πίνακας αποτελεσμάτων εκπαίδευσης

6) Pre-trained Neural Networks

In order to optimize the prediction results, we tried to modify pre-trained neural networks on the ImageNet [11] dataset using the transfer learning method. By transfer learning we mean that we keep the result of the already existing training that has been carried out in a neural network (in our case on ImageNet which means that it has been trained to recognize 1000 categories of images) and then we add some layers to specify the result of learning in our problem. Finally, we fit our data into this neural combo. The architecture below, was used for the training of neural networks with Transfer Learning:

1. First of all, we load the neural network we want and determine that we want it to use the weights that have resulted from its training on the ImageNet dataset.
2. Furthermore, we flatten the result that will arise out of the pre-trained neural networks in order to advance it to the fully connected layers.
3. Finally, we created 3 Fully connected dense layers of which the first and the second have 256 neurals while the last one has 4 neurals (as many as the class we want to predict).



Transfer Learning neural network architecture diagram.

At this stage of the work, various pre-trained neural networks were tested which were trained for 30 seasons in the training data. In the table below we see in detail the results that emerged from the training.

Model	Val Accuracy	Training time (sec)	Total Epoch
VGG16	0,9231	830,84	30
MobileNetV2	0,9223	675,34	30
DenseNet169	0,9134	928,5	30
InceptionResNetV2	0,9112	951,97	30
DenseNet201	0,9093	1031,09	30
InceptionV3	0,9025	769,97	30
VGG19	0,8987	879,63	30
ResNet50	0,7431	755,7	30

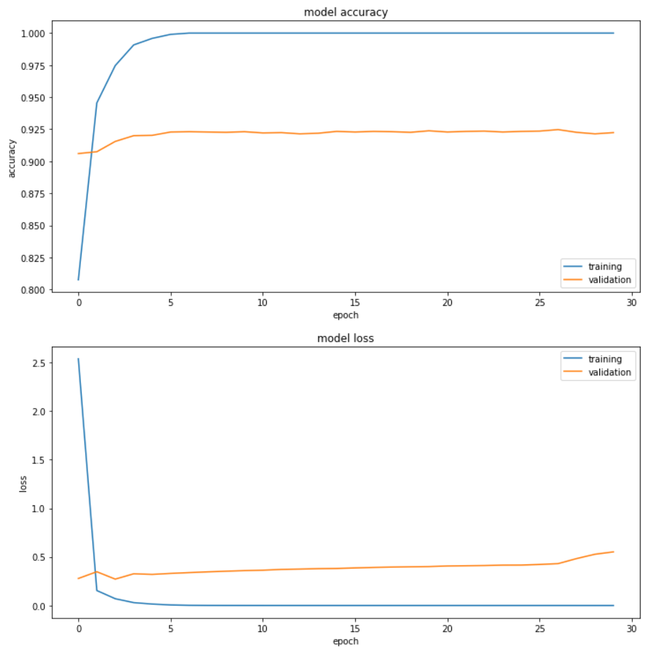
Transfer Learning Neural Network Scoreboard

We observe that the best performance was achieved by the neural networks VGG16, MobileNetV2 and DenseNet169 with higher accuracy to achieve for the VGG16 with 92.31%.

7) Selection of models for Mobile App

To create the Mobile Application, we decided to export and test the 3 models that presented the highest results. The following is a brief description of these models.

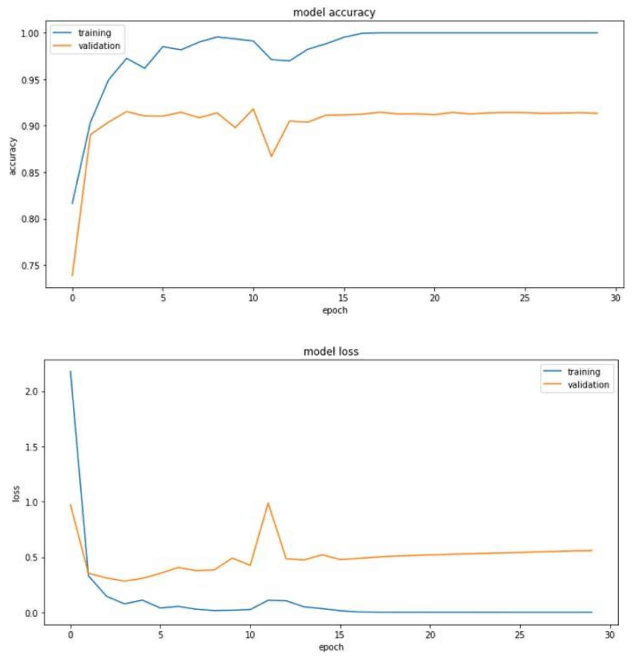
Mobile Net: As its name suggests, the MobileNet model [24] is designed to be used in mobile applications and is the first computer vision model for mobile devices. MobileNet uses depthwise seperable convolutions. Significantly reduces the number of parameters compared to neural networks that use simple convolutions and have the same depth. This results in a "lighter" deep neural network.



Accuracy and Loss line charts for MobileNetV2

From the graphs above, we observe that the performance of the model is stable from the very first seasons, also the loss seems to be stable. Therefore, we conclude that the model could have stopped training in 10 to 15 epochs to avoid overfitting.

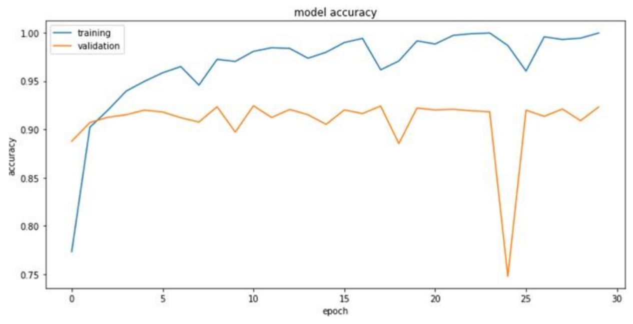
DenseNet-169: The densenet-169 [25] model, is one of the DenseNet models designed for image classification. The main difference with the densenet-121 model is the size and accuracy of the model. The densenet-169 is about 55MB bigger in size compared to the approximately 31MB size of the densenet-121 model. All DenseNet models are pre-trained in the ImageNet image database

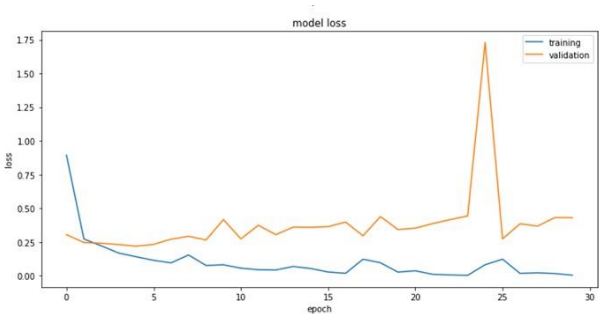


Accuracy και loss line charts for densenet-169

From the graphs above, we observe that the performance of the model stabilizes from epoch 15 onwards, also the validation loss increases from epoch 15 onwards. We therefore conclude that the model could have stopped training at 15 seasons to avoid any overfitting.

VGG16: VGG16 [13] (also called OxfordNet) is a convolutional neural network architecture named after the Visual Geometry Group that developed it. It was used to win the ILSVRC2014 (Large Scale Visual Recognition Challenge 2014) in 2014. It was still considered an excellent vision model. The VGG-16 is a convolutional neural network with a depth of 16 layers. The model loads a set of weights pre-trained in ImageNet. The model achieves 92.7% of the top 5 accuracy tests on ImageNet VGG16 [13].





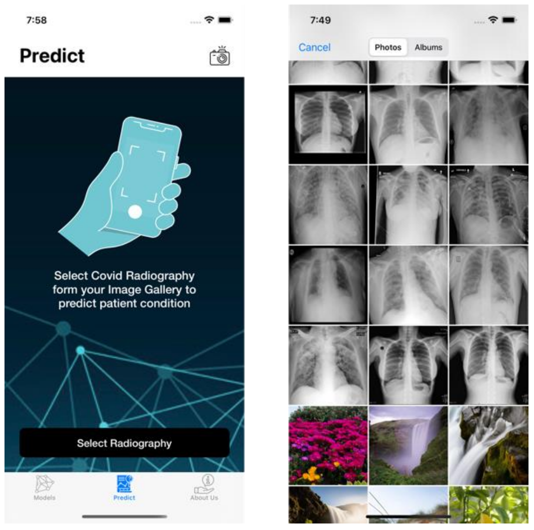
Accuracy και loss line chart for VGG16

From the graphs above, we notice that the performance of the model has not stabilized yet, so it may take more epochs for this particular model.

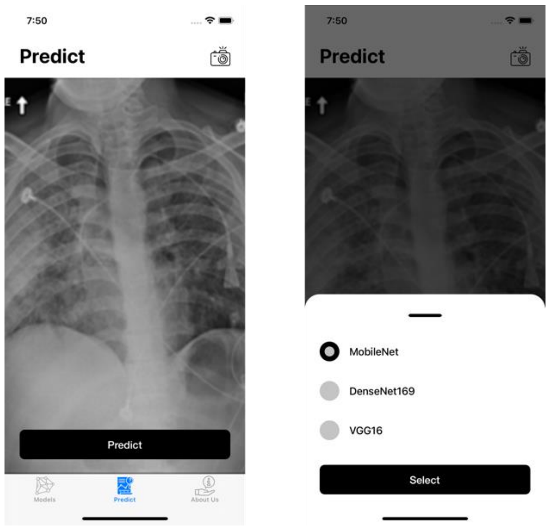
8) Description of Mobile Application

The Mobile Application created in the framework of the application for the utilization of the machine learning models that we built targets for iOS devices that support software newer than iOS 11. For the implementation of the application the native tools provided by Apple were used and more specifically for the management of neural networks, the CoreML framework was used [26]. The CoreML framework enables us to load and use neural network models in our application in the form of “.mlmodel” files. To convert the ".m5" files to “.mlmodel” an appropriate script was configured that performs this process using the coremltools package. The following is a description of the user interface of the application.

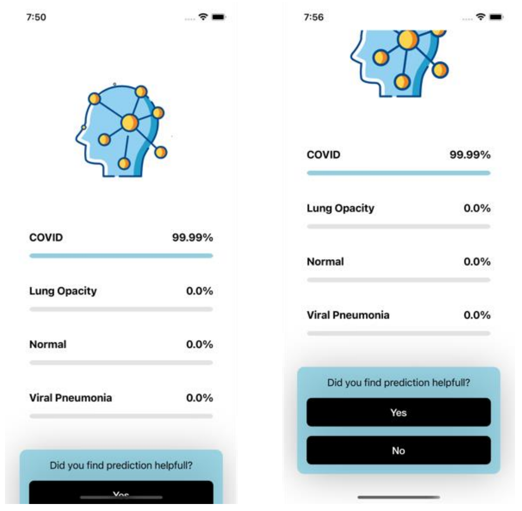
Home Page: On this page the user has the opportunity to select the X-ray image he wants from the mobile gallery.


7:58
Predict
Select Covid Radiography form your Image Gallery to predict patient condition
Select Radiography
Models
Predict
About Us
7:49
Cancel
Photos
Albums

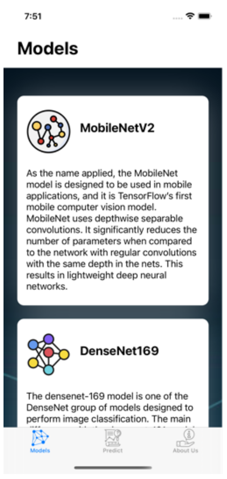
Neural Network Selection: After, the user has selected the X-ray in which he wants to make a diagnosis, by clicking the predict button, a dialog is displayed in which he selects the neural network he wants to use.



Prediction Results: Then, performing the diagnosis for the specific X-ray, the probability / classification predicted by the algorithm is displayed on the screen.



Model Description Page: The following page provides a brief description of the models we used.


7:51

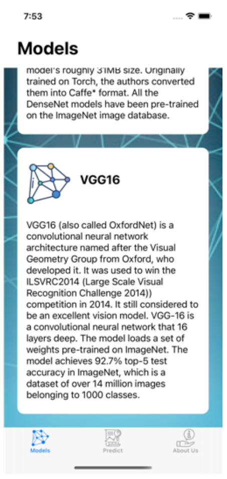
Models

MobileNetV2

As the name applied, the MobileNet model is designed to be used in mobile applications, and it is TensorFlow's first mobile computer vision model. MobileNet uses depthwise separable convolutions. It significantly reduces the number of parameters when compared to the network with regular convolutions with the same depth in the nets. This results in lightweight deep neural networks.

DenseNet169

The densenet-169 model is one of the DenseNet group of models designed to perform image classification. The main


7:53

Models

model's roughly 31MB size. Originally trained on Torch, the authors converted them into Caffe* format. All the DenseNet models have been pre-trained on the ImageNet image database.

VGG16

VGG16 (also called OxfordNet) is a convolutional neural network architecture named after the Visual Geometry Group from Oxford, who developed it. It was used to win the ILSVRC2014 (Large Scale Visual Recognition Challenge 2014)) competition in 2014. It still considered to be an excellent vision model. VGG-16 is a convolutional neural network that 16 layers deep. The model loads a set of weights pre-trained on ImageNet. The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes.

9) Conclusions and Future Extensions

After studying, constructing and implementing the deep algorithms we created we conclude that the problem can be modeled very well with the help of pre-trained neural networks which provide quite satisfactory predictions. As future extensions we would like to study our problem in new data and consider the possibility of improving performance. From the user interface part, the next goal is to build a cross platform application that will provide the above functions on both iOS and Android devices.

10) Βιβλιογραφία

1. World Health Organization Health Topics / Coronovirus
Available at:
https://www.who.int/health-topics/coronavirus#tab=tab_1

2. Kathy Katella Comparing the COVID-19 Vaccines: How Are They Different?
Available at:
https://www.yalemedicine.org/news/covid-19-vaccine-comparison

3. Information on Rapid Molecular Assays, RT-PCR, and other Molecular Assays for Diagnosis of Influenza Virus Infection
Available at:
https://www.cdc.gov/flu/professionals/diagnosis/molecular-assays.htm

4. mane Abdelli In silico study the inhibition of angiotensin converting enzyme 2 receptor of COVID-19 by Ammoides verticillata components harvested from Western Algeria Available at:
https://pubmed.ncbi.nlm.nih.gov/32362217/

5. WHAT IS THE DIFFERENCE BETWEEN RAPID ANTIGEN TESTS, MOLECULAR RT-PCR TESTS AND ANTIBODY TESTS?
Available at:
https://nipd.com/articles/blog/whats-the-difference-between- rapid-antigen-tests-molecular-rt-pcr-tests-and-antibody-tests/

6. Ilija Mihajlovic Everything You Ever Wanted To Know About Computer Vision.
Available at:
https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e

7. Sumit Saha A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way
Available at:
https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

8. Saad Awadh Alanaz Boosting Breast Cancer Detection Using Convolutional Neural Network
Available at:
https://www.hindawi.com/journals/jhe/2021/5528622/

9. Mrudul Shah How to Apply Machine Learning (ML) in an Android App
Available at:
https://towardsdatascience.com/how-to-apply-machine-learning-ml-in-an-android-app-33e848c0dde6

10. What Is Image Recognition?Available at:
https://medium.com/dataman-in-ai/module-6-image-recognition-for-insurance-claim-handling-part-i-a338d16c9de0

11. ImageNet Large Scale Visual Recognition Challenge (ILSVRC)
Available at:
https://www.image-net.org/challenges/LSVRC/

12. Alex Krizhevsky ImageNet Classification with Deep

Convolutional Neural Networks
Available at:
https://papers.nips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

13. Karen Simonyan & Andrew Zisserman VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION
Available at:
https://arxiv.org/pdf/1409.1556.pdf

14. Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun Deep Residual Learning for Image Recognition
Available at:
https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf

15. Saad Awadh Alanaz Boosting Breast Cancer Detection Using Convolutional Neural Network
Available at:
https://www.hindawi.com/journals/jhe/2021/5528622/

16. James D. Dormer Convolutional Neural Networks for the Detection of Diseased Hearts Using CT Images and Left Atrium Patches
Available at:
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6123226/

17. Fuzhen Zhuang A Comprehensive Survey on Transfer Learning
Available at:
https://arxiv.org/abs/1911.02685

18. Narin A., Kaya G., & Pamuk Z. (2020). Automatic detection of Coronavirus disease (COVID-19) using X-ray images and deep convolutional neural networks.
Available at:
https://link.springer.com/article/10.1007/s10044-021-00984-y

19. Christian Szegedy Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning
Available at:
https://arxiv.org/abs/1602.07261v2

20. Kaggle dataset COVID-19 Radiography Dataset
Available at:
https://www.kaggle.com/tawsifurrahman/covid19-radiography-database

21. What is Colaboratory?
Available at:
https://colab.research.google.com/notebooks/intro.ipynb?utm_source=scs-index

22. Tensor Flow Core Use TPUs
Available at:
https://www.tensorflow.org/guide/tpu

23. Convolution Wikipedia
Available at:
https://en.wikipedia.org/wiki/Convolution

24. Mark Sandler, Andrew Howard MobileNetV2: Inverted Residuals and Linear Bottlenecks
Available at:
https://arxiv.org/abs/1801.04381

25. Gao Huang, Zhuang Liu,Densely Connected Convolutional Networks
Available at:
https://arxiv.org/abs/1608.06993

26. CoreML
Available at:
https://developer.apple.com/documentation/coreml