CBS final project

Kacper Domżał

June 28, 2021

Abstract

The aim of my project was to build a classifier using deep learing that would decide whether someone has pneumonia or not by analyzing chest X ray. I thought it would be an interesting project due to the fact that because of the current pandemic many chest X ray are being made as a part of COVID-19 diagnosis. The data set I chose for this task comes from kaggle - 5856 chest Xray photos, mostly of people with pneumonia (pneumonia:healthy lungs are in around 3:1 ratio).

To achieve my task I decided to use tensorflow library. I build 2 classifiers in total. The first one is a pretty simple convolutional network build by hand and trained from scratch. The second model is based on a already pretrained and constructed DenseNet169 network - I decided to try transfer learning. DenseNet169 has 169 layers as the name suggest, which is a lot compared to the first network I created. My goal was to try and find out by how much a complicated deep learning network such as DenseNet169 outperforms a simple network such as my own convolutional network.

All the parameters used in networks were chosen by trial and error in order to achieve the best performing classifier. Unfortunately, I did not have enough computational resources to perform some broad optimal parameters search in loops etc. so I did it mostly by hand.

1 Introduction

Nowadays, during the coronavirus pandemic, millions of people suffer from health complications related to COVID-19. Even though the COVID-19 is a new virus, not yet very well described and characterized, after 2 years of pandemic scientist and doctors have been able to find some patters and properties of viral infection caused by coronavirus. Most of the people infected by coronavirus recover fully with no complications, but unfortunately many people infected by coronavirus do suffer from a wide range of complications. One of those complications is a viral pneumonia. A very popular and a good screening diagnostics of a viral pneumonia is a chest X ray. Chest X ray does not always give conclusive results and does not allow to differentiate between viral and bacterial pneumonia but it is fast, cheap and basically almost all hospitals have the equipment needed to do chest X rays [RLJM11]. Because of that, a classifier that would decide whether someone has pneumonia could be useful. Also, because of the millions of chest X rays that are being made now, such classifier could be trained on a big and divers data set. Such classifier could be useful in some bulk chest X ray analysis, could be used as a part of some decision tree that would aim to optimize COVID-19 treatment strategies or a decision tree that would try to predict the outcome of an infection and possible after COVID-19 complications.

2 Materials and Methods

In this section the technical site of my project will be explained.

2.1 Materials

The data set used to prototype classifier was obtained from Kaggle. It contains 5856 chest X ray images divided into 3 groups:

1. Healthy lungs
2. Viral pneumonia
3. Bacterial pneumonia

Because of the fact that it is very difficult to tell if someone has a viral pneumonia or a bacterial pneumonia (sometimes it is even impossible [RLJM11]), I decided to merge these two groups together, so that my final data set contains only 2 groups:

1. Healthy lungs
2. Pneumonia

Then, all then chest X ray images were divided into 3 sets:

1. Training set - around 85 percent of all images
2. Validation set - around 10 percent of all images
3. Test set - around 5 percent of all images

It is worth mentioning that the original data set is not very well balanced [Fig1]. The proportions of healthy lungs to pneumonia is about 1:3. The cause of that seams to be pretty easy to explain - healthy people with no symptoms of pneumonia do not usually get a chest X ray. The reason for that is X ray imaging is not very precise, there are better, more precise and emitting less radiation methods that are used for imaging for medical purposes, like Nuclear magnetic resonance spectroscopy and computed tomography scan. I decided to up the proportions of healthy lungs to pneumonia in the validation set to prevent the classifier from classifying all images as pneumonia [Fig1], due to the disproportions in the data set.

2.2 Methods

I decided to create two deep learning classifiers, both based on convolutional networks. First network that I created was a pretty basic convolutional network [Fig2]. The network had 11 layers and around 2 million parameters. The second network was based on a already trained convolutional network - DenseNet169 [Fig3]. The DenseNet169 as the name suggests has 169 which were not trained on the data set I obtained from kaggle. To this network I added 10 layers that were trained on the X ray data set.

The overall goal behind creating two classifiers so different in complexity was to find out how the additional layers and the use of transfer learning influences the performance of the classifier.

Because of the class imbalance in the data set (pneumonia : healthy lungs) data augmentation had to be used, otherwise classifiers would predict all images to be pneumonia. The used data augmentation parameters and functions can be seen in the jupyter notebook script located in the same directory.

Due to the fact I had limited computational resources I was not able to perform large scale search for the best parameters/networks. I found the configuration that can be seen here by trail and error, while trying to keep the networks decently simple to speed up the training process and avoid over fitting. To keep the accuracy on the test set higher I used early stopping and the reduce learning rate on plateau technique.

The implementations of the classifiers was done in python 3.8. The networks were created using tensorflow library (version 2.5.0). The images fed to the classifiers were RGB, trying to convert them to black and white images would result in worsening the network performance on the test set.

3 Results

Both classifiers created by me achieved satisfactory accuracy on the test set. My first model that has 10 layers achieved 84 percent accuracy and the modified DenseNet169 that has 179 layers got 89 percent accuracy. Such small difference between models so different suggest the deciding whether someone has pneumonia based on their chest X ray is pretty straight forward, and it pretty much is, in most cases.

As you can see on the images [Fig4, Fig5] sometimes deciding whether someone has pneumonia is difficult, it's not visible at first glance, sometimes it's not visible for me at all. Hence, the difference in performance. The more complex model seems to be able to learn some properties from the data set that are not comprehensible to most humans. It is also worth to take into consideration that pneumonia vary from case to case. Some cases are severe and the chest X ray chest images are pretty easy to classify and some are milder, for example when the pneumonia was caught early and the infection has not spread across lungs so much.

4 Discussion

The accuracy of the model created using transfer learning seems decent, considering the small data set as for deep learning. As I mentioned earlier, pneumonia is rather easy to recognize in most cases (by looking at the chest X ray), but in some cases it's very difficult. Different X ray machines generate different looking images, some people have titanium plates in their spines, men and women are build differently. Those are only some of the problems that the classifier has to overcome to improve it's accuracy. The most reasonable way to improve classifiers performance is to gather more chest X ray images and train networks on a bigger data set. Using a large, diverse data set could and probably would, lead to creating a classifier with a test set accuracy of over 90 percent. Hopefully, the chest X rays that are being made now as a part of COVID-19 screening will be stored and will be publicly accessible in the future so that a accurate classifier could be made.

5 Graphics

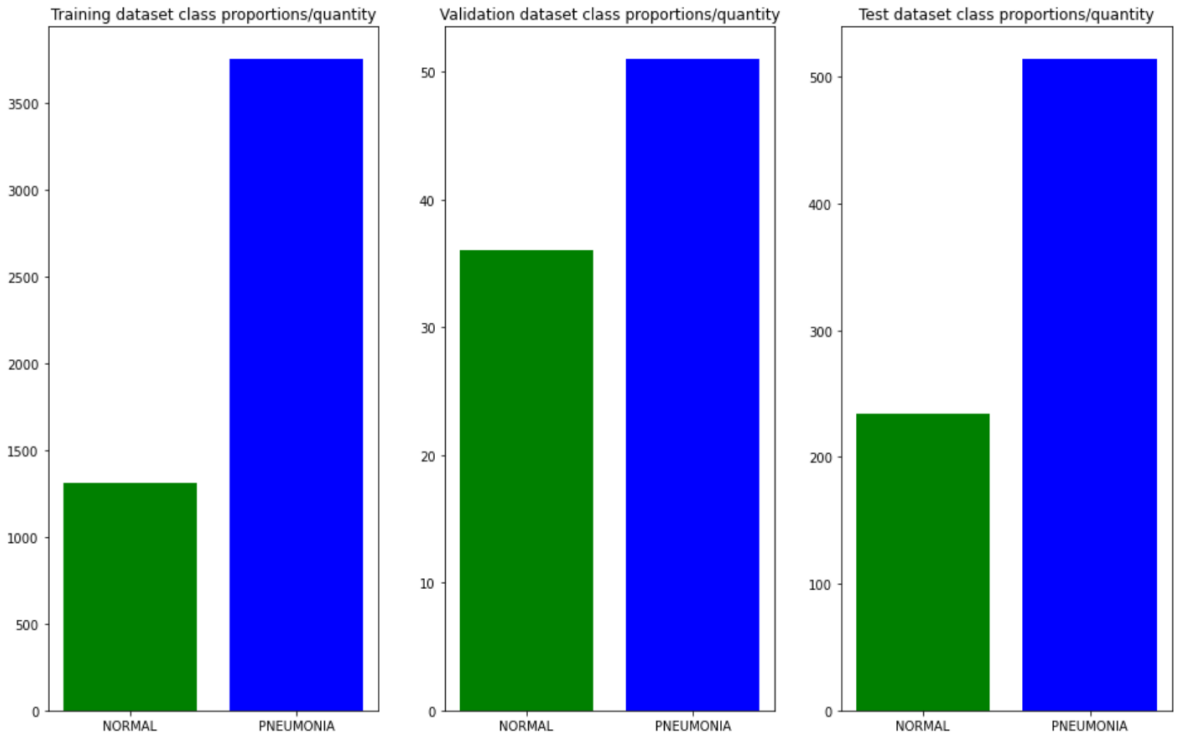


Fig1. Data set division into: training, validation and test sets.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 198, 198, 64)      1792
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 99, 99, 64)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 97, 97, 64)        36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 48, 48, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 46, 46, 64)        36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 23, 23, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 33856)             0
_________________________________________________________________
dense (Dense)                (None, 64)                2166848
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 65
=================================================================
Total params: 2,242,561
Trainable params: 2,242,561
Non-trainable params: 0
```

Fig2. First network summary.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
densenet169 (Functional)     (None, 6, 6, 1664)        12642880
_________________________________________________________________
global_average_pooling2d (Gl (None, 1664)              0
_________________________________________________________________
batch_normalization (BatchNo (None, 1664)              6656
_________________________________________________________________
dense_2 (Dense)              (None, 256)               426240
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0
_________________________________________________________________
batch_normalization_1 (Batch (None, 256)               1024
_________________________________________________________________
dense_3 (Dense)              (None, 128)               32896
_________________________________________________________________
dropout_2 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_4 (Dense)              (None, 64)                8256
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 65
=================================================================
Total params: 13,118,017
Trainable params: 471,297
Non-trainable params: 12,646,720
```

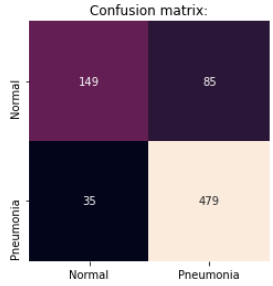
Fig3. Second network based on DenseNet169 summary.

```
Classification report:
                  precision    recall  f1-score   support

           0       0.81      0.64      0.71       234
           1       0.85      0.93      0.89       514

    accuracy                           0.84       748
   macro avg       0.83      0.78      0.80       748
weighted avg       0.84      0.84      0.83       748

Confusion matrix:
```



Wrong classification examples:

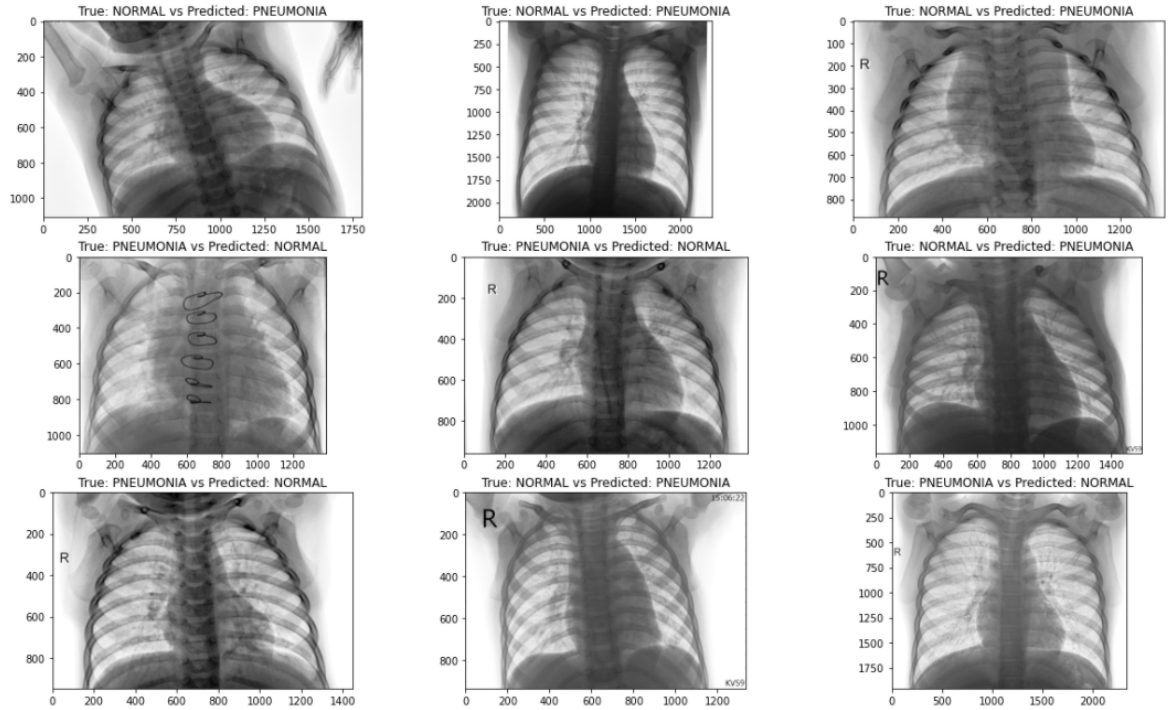


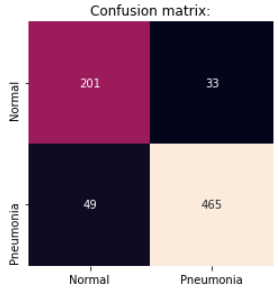
Fig4. Performance summary of the first network.

```
Classification report:
                    precision    recall  f1-score   support

           0       0.80      0.86      0.83       234
           1       0.93      0.90      0.92       514

    accuracy                           0.89       748
   macro avg       0.87      0.88      0.87       748
weighted avg       0.89      0.89      0.89       748

Confusion matrix:
```



Wrong classification examples:

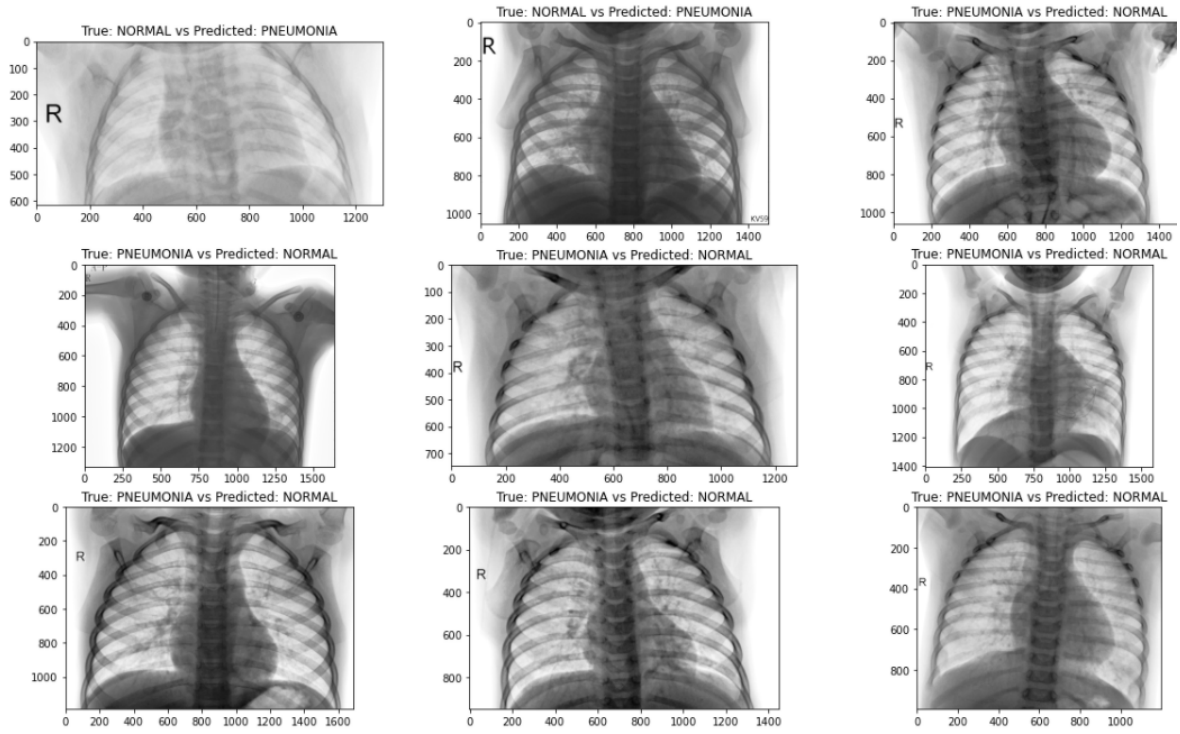


Fig5. Performance summary of the second network.

References

[RLJM11] Olli Ruuskanen, Elina Lahti, Lance C Jennings, and David R Murdoch. Viral pneumonia. *The Lancet*, 377(9773):1264–1275, 2011.