WB COVID NET

March 2021

1 Description of the dataset

The dataset for COVID-Net was combined from 3 repositories and two kaggle database, gathering collectively 16690 examples until today(17.03.2021). From what is written we can deduce that all samples are the X-rays photos. Moreover, each one is in JPG format and on a gray scale. Unfortunately every photo has different size. However, we know the total wieght of the dataset which is 6 874 194 718 B $\cdot 8 = 54\ 993\ 557\ 744$ b. Thus we can compute that a single photo has around 2 878 189 b. The analyzed model mainly focused on recognized the 3 categories of the lungs state: normal, pneumonia and Covid-19 and took photos of size 480×480 as an input, although their original size was 1024×1024

2 Dataset EDA - each class size and associated risks

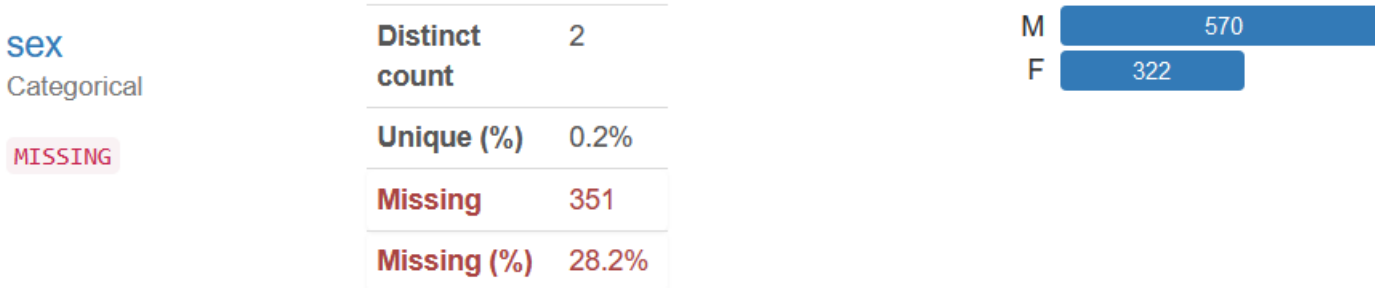
Firstly lack of the public data for COVID cases was emphasized in the article, however, since the latest update in the article, the situation has improved a bit.

> comes available to improve the dataset. More specifically, the COVIDx dataset contains 183 CXR images from 121 COVID-19 patient cases. For CXR images with no pneumonia and non-COVID19 pneumonia, there are significantly more patient cases and corresponding CXR images. More specifically, there are a total of 8,066 patient cases who have no pneumonia (i.e., normal) and 5,538 patient cases who have non-COVID19 pneumonia. Dataset

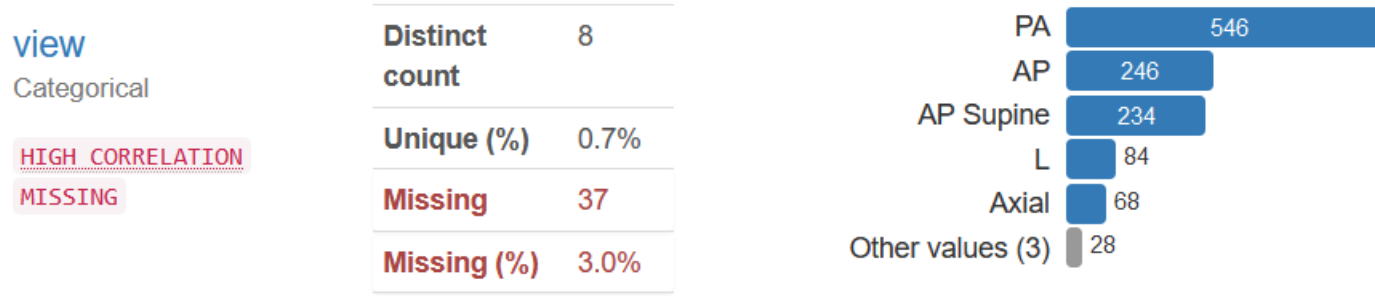
For the time being COVIDx dataset can be summarized with output produced by jupyter notebook intended to combine data from various sources.

```
Final stats
Train count:  {'normal': 7966, 'pneumonia': 5475, 'COVID-19': 1670}
Test count:  {'normal': 885, 'pneumonia': 594, 'COVID-19': 100}
Total length of train:  15111
Total length of test:  1579
```

Metadata corresponding to the extracted samples is inconsistent and missing values to the point that the imputation or any meaningful analysis on the majority of parameters cannot be performed, Only two parameters viable for analysis are sex and view (describing the type of photo).



We can observe the significant inbalance, however to the best of our knwoledge it should not impact the training process in any meaningful way.



The distribution of data between the most common types of view to the best of our knowledge appears to be sufficient for the accurate training process.

All these factors may have negative influence on the performance of final network. Insufficient number of data may get in the way of acquiring model with great performance. It was also pointed out in the article that sensitivity for COVID cases was limited.

scenarios. First, it can be observed that COVID-Net can achieve good sensitivity for COVID-19 cases (87.1% sensitivity), which is important since we want to limit the number of missed COVID-19 cases as much as possible. While promising, it should be noted that the number of COVID-19 patient cases available is limited compared to the other infection types in COVIDx and as such a better view of effectiveness will improve as more COVID-19 patient cases becomes available. Second, it can be observed that COVID-

Too little data may result in poor performance due to over-constrained model underfitting or under-constrained model likely ovefitting to the training dataset.

When it comes to databalance, noteworthy is the fact that the model may achieve good performance for wrong reasons, by prioritizing larger class. Such error is likely to occure when accuracy is used as a metric to evaluate the model performance. This brings us to the next section - ways to deal with unbalanced datasets.

3 Implemented methods to deal with unbalanced dataset

- As mentioned in paragraph above the most basic thing is to use appropriate metrics; for example those can be sensitivity and PPV which have been chosen by COVID-Net contributors
- Another basic thing is to reduce imbalance as much as possible by combining more available datasets
- In order to gain optimal solution, first steps can be taken during the Architecture Design stage. It is crucial for COVID-Net to limit the number of missed COVID-19 cases, therefore additional conditions to machine-driven design exploration have been set

  human specified design requirements. More specifically, the following human specified design requirements were employed in this study to enable the generative synthesis process to learn and identify the optimal macroarchitecture and microarchitecture designs for the final COVID-Net network architecture: (i) COVID-19 sensitivity $\geq$ 80%, and (ii) COVID-19 positive predictive value (PPV) $\geq$ 80%.

- data augmentation

  of epochs=22, batch size=8, factor=0.7, patience=5. Furthermore, data augmentation was leveraged with the following augmentation types: translation, rotation, horizontal flip, and intensity shift. Finally, we introduce a batch

Each of those methods results in acquiring larger set of photos.

- batch re-balancing

> tal flip, and intensity shift. Finally, we introduce a batch re-balancing strategy to promote better distribution of each infection type at a batch level. The initial COVID-Net pro-

This method is intended to prevent model to fit to the data distribution.

Implementation of methods mentioned above can be found in data.py script available in repository.

For example, let's take a brief look at the following code

```
_augmentation_transform = ImageDataGenerator(
    featurewise_center=False,
    featurewise_std_normalization=False,
    rotation_range=10,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=True,
    brightness_range=(0.9, 1.1),
    zoom_range=(0.85, 1.15),
    fill_mode='constant',
    cval=0.,
)
```

As one may notice, the data augmentation was performed via ImageDataGenerator imported from Keras library with usage of TensorFlow backend.

4 Random over- and undersampling for data distribution adjustment

To perform over- and undersampling we have created new labels, where depending on a method type we have randomly selected images to remove or to duplicate them for model to train on. We present the results below (that will be for the first and 10th epoch - 10 is the default number in provided by researchers script).

Let's take a look at the training results for data augmentation proposed by COVID-NET creators:

```
2689/2689 [==============================] - 2331s 862ms/step
Epoch: 0001 Minibatch loss= 1.092139840
[[100.   0.   0.]
 [ 26.  70.   4.]
 [ 11.   2.  87.]]
Sens Normal: 1.000, Pneumonia: 0.700, COVID-19: 0.870
PPV Normal: 0.730, Pneumonia 0.972, COVID-19: 0.956
```

```
Epoch: 0010 Minibatch loss= 0.602773011
[[97.  3.  0.]
 [ 6. 90.  4.]
 [ 9.  2. 89.]]
Sens Normal: 0.970, Pneumonia: 0.900, COVID-19: 0.890
PPV Normal: 0.866, Pneumonia 0.947, COVID-19: 0.957
Saving checkpoint at epoch 10
Optimization Finished!
```

Results for performed undersampling:

```
Epoch: 0001 Minibatch loss= 0.310275257
[[98.  1.  1.]
 [18. 80.  2.]
 [ 7. 20. 73.]]
Sens Normal: 0.980, Pneumonia: 0.800, COVID-19: 0.730
PPV Normal: 0.797, Pneumonia 0.792, COVID-19: 0.961
```

```
Epoch: 0010 Minibatch loss= 0.142740890
[[74. 24.  2.]
 [ 4. 94.  2.]
 [ 2. 11. 87.]]
Sens Normal: 0.740, Pneumonia: 0.940, COVID-19: 0.870
PPV Normal: 0.925, Pneumonia 0.729, COVID-19: 0.956
Saving checkpoint at epoch 10
Optimization Finished!
```

Results for oversampling - unfortunately we have results for the first epoch so far. We are aware that better analysis would be performed for 10th epoch and if it is possible we will provide results in later homeworks or commit them separately.

```
Epoch: 0001 Minibatch loss= 1.465883493
[[99.  1.  0.]
 [13. 86.  1.]
 [ 8.  4. 88.]]
Sens Normal: 0.990, Pneumonia: 0.860, COVID-19: 0.880
PPV Normal: 0.825, Pneumonia 0.945, COVID-19: 0.989
Saving checkpoint at epoch 1
```

Size of each class for undersampling: 517 Size for oversampling: COVID - 5222; Normal - 7966; non-COVID pneumonia - 5423

Conclusions:
There is a noticable difference in value of minibatch loss function. The less data is provided for the model to learn the easiest it is for him to fit the data. Naive methods of data augmentation do not provide as good results. Metrics seem to present worse values and not as stable as one may expect. (Few of values are significantly worse than the others.

Short script used used to create appropirate .txt label files is attached.

In this .pdf file we do not describe the steps we took in order to run training code as it is not mentioned in homework requirements, however, we can provide it any time if needed.