Supplementary Materials

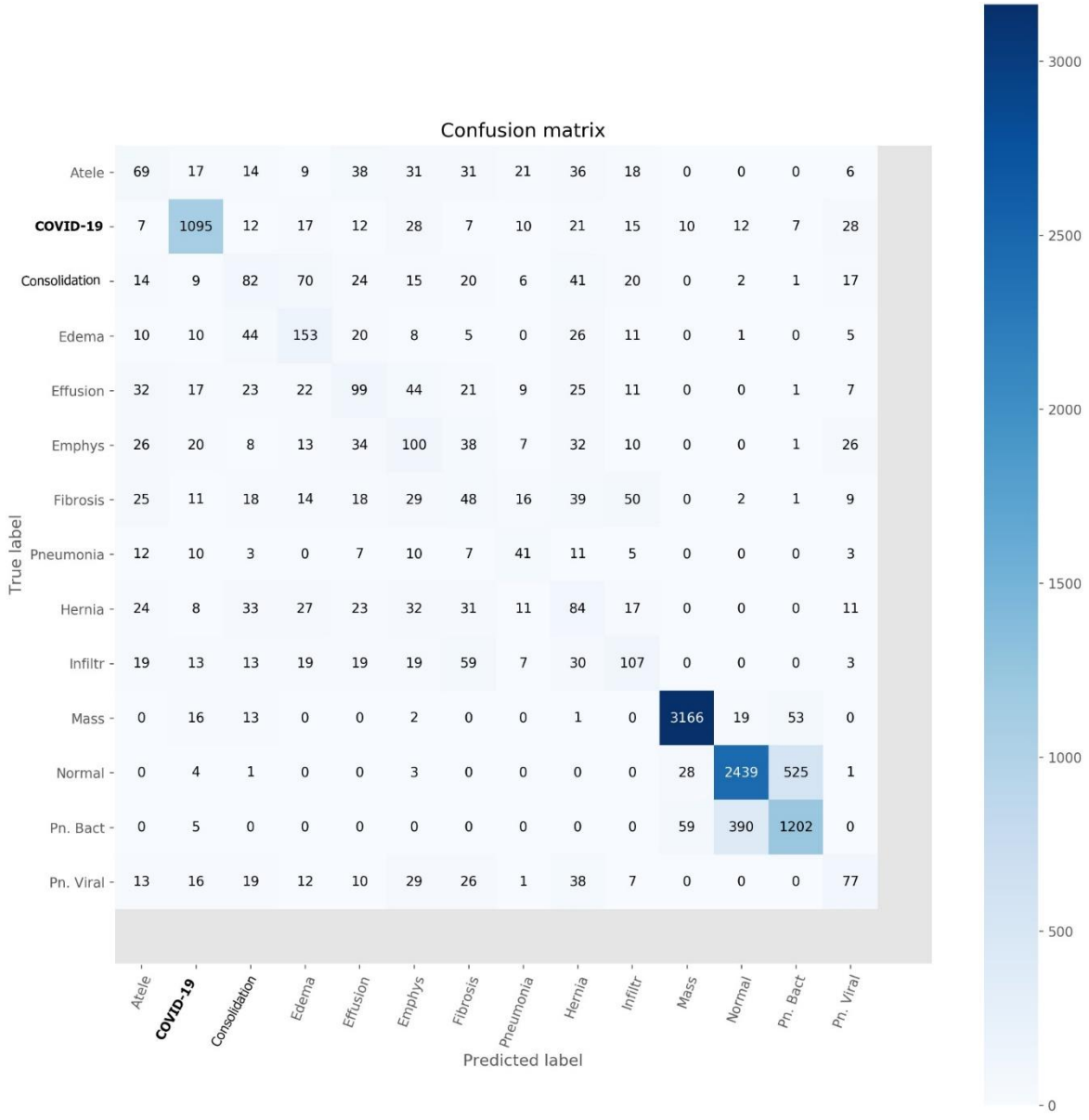

Figure S1. Confusion Matrix for the Multiclass dataset.

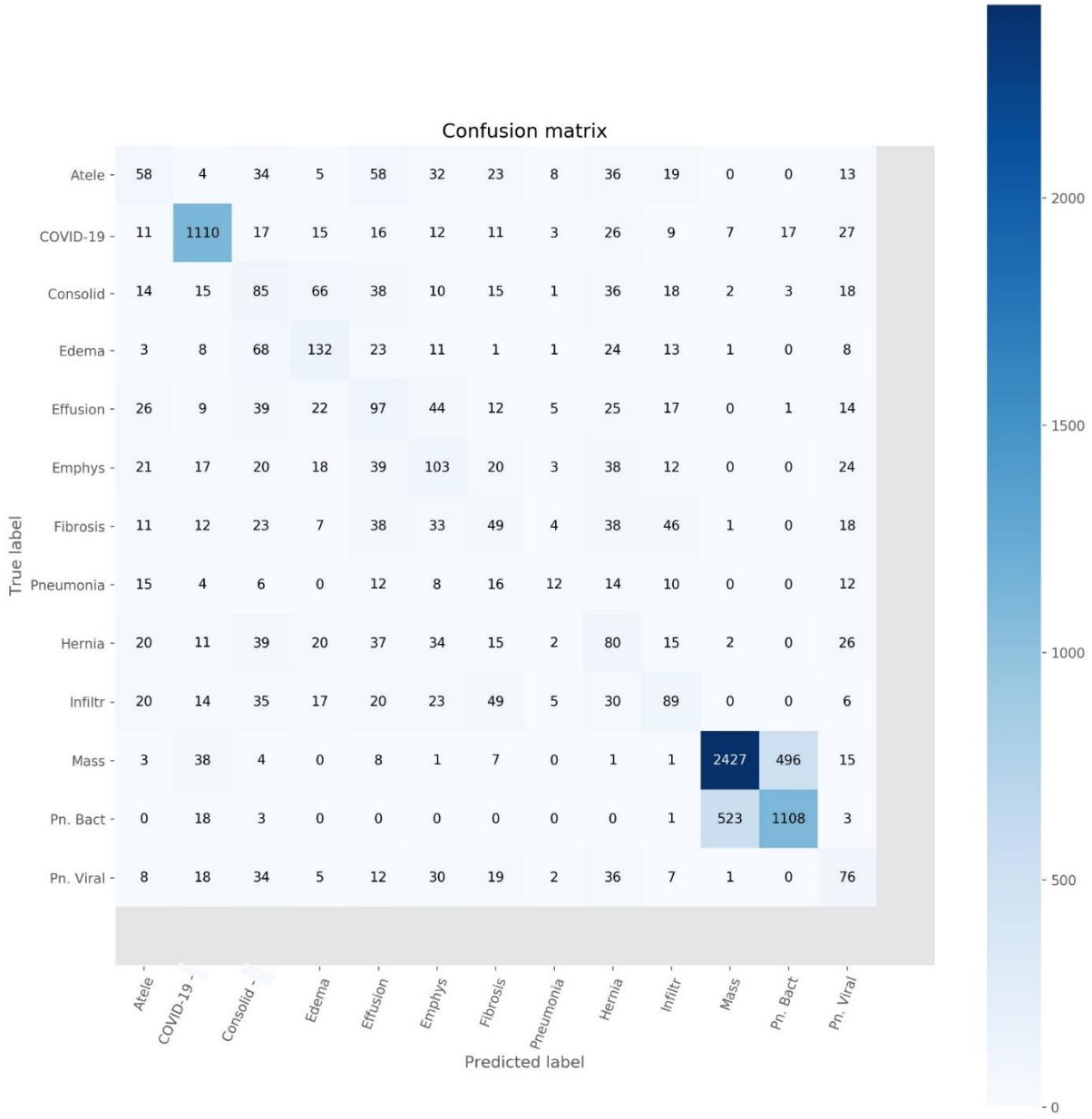

Figure S2. Confusion Matrix for the Abnormality discrimination dataset.

MobileNet (v2) Parameters and Python implementation

1. Parameters

Table S1. MobileNet (v2) parameters and hyper-parameters.

Version	2
Input Shape (height,width,channels)	400x400x1
Last Convolution path output name	`'conv_pw_13_relu'`
Global Pooling	Global Average Pooling after the last activation
Classification network	One densely connected layer (2500 nodes) and one layer of two nodes for binary classification
Loss Function	Categorical Cross-Entropy
Optimizer	Adam with the default parameters of tf.keras.optimizers
Extra Batch Normalization layers	Yes, inside the densely connected layer at the top of the network
Dropout layer	Yes, 50% dropout inside the densely connected layer at the top of the network
Epochs	40
Batch Size	32

2. Python code

Python version: 3.8
Tensorflow version: 2.0

```
def make_mobile (in_shape, tune, classes):

    base_model = tf.keras.applications.MobileNet(
    include_top=False,
    weights="imagenet",
    input_tensor=None,
    input_shape=in_shape,
    pooling=None,
    classes=classes)

    layer_dict = dict([(layer.name, layer) for layer in base_model.layers])

    for layer in base_model.layers:
        layer.trainable = True #or False, depending on the experiment

    x1 = layer_dict['conv_pw_13_relu'].output
    x1= tf.keras.layers.GlobalAveragePooling2D()(x1)
```

```
    x = tf.keras.layers.Dense(2500, activation='relu')(x1)
    x = tf.keras.layers.BatchNormalization()(x)
    x = tf.keras.layers.Dropout(0.5)(x)
    x = tf.keras.layers.Dense(classes, activation='softmax')(x)
    model = tf.keras.Model(inputs=base_model.input, outputs=x)

    model.summary()
    model.compile(optimizer='adam' loss='categorical_crossentropy',
metrics=['accuracy'])
    plot_model(model, to_file='mobile.png')
    print("[INFO] Model Compiled!")
    return model
```