Assignment Proposal Report
On

IMAGE CLASSIFICATION

Submitted by

ANKIT SRIVASTAVA (212IS003)
KAUSHAL KISHOR (212IS013)
JATIN CHANDRA (212IS011)

Dept. of Computer Science and Engineering

NITK, Surathkal

Date submission: 31-12-2021

DL – ASSIGNMENT: 1

AIM: To create image classification model based on given data set.

Dataset: Dataset is divided into three parts train set, validation set, test set and each folder contains images of three classes cap, cov and norm.

1. Cap: Images of phenomena class.
2. Cov: Images of covid class.
3. Norm: Images of normal class.

Models used: Initially we have used different model like LeNet, AlexNet, VGG etc. In all these case we achieved train accuracy of ~0.8 to 0.9 but we are not able to get a good validation accuracy, we achieved only ~0.52 to ~0.58 validation accuracy. After that we have used ResNet50 with different technique and able to get to final result.

Final model: ResNet50 is used with 1 dropout of 0.2 and 'adam' is used as optimizer. Here we have used class weights as well because data is imbalance. We run this model of 25 epochs with batch size of 32. Loss function used is categorical cross entropy. (256x 256 x 3) Input size RGB images are used.

Implementation:

1. **Libraries used:**

```
import tensorflow as tf
from keras.preprocessing.image import ImageDataGenerator
import numpy as np
import cv2
import matplotlib.pyplot as plt
import keras
from tensorflow.keras.utils import to_categorical
from keras import models
from keras import layers
import collections
from keras.callbacks import ModelCheckpoint,CSVLogger
from sklearn.metrics import classification_report,confusion_matrix
import itertools
```

2. Loading data:

Extracting Train data

```
train_datagen = ImageDataGenerator( rescale=1./255,
                                    shear_range=0.3,
                                    zoom_range=0.2)
train_set = train_datagen.flow_from_directory(
        'data/Train',
        target_size=(256, 256),
        batch_size=32,
        class_mode='categorical',
        shuffle=True)
```

```
Found 5349 images belonging to 3 classes.
```

Extracting validation data

```
[5] valid_datagen = ImageDataGenerator(rescale=1./255)
    valid_set = valid_datagen.flow_from_directory(
            'data/Valid',
            target_size=(256, 256),
            batch_size=32,
            class_mode='categorical',
            shuffle=True)
```

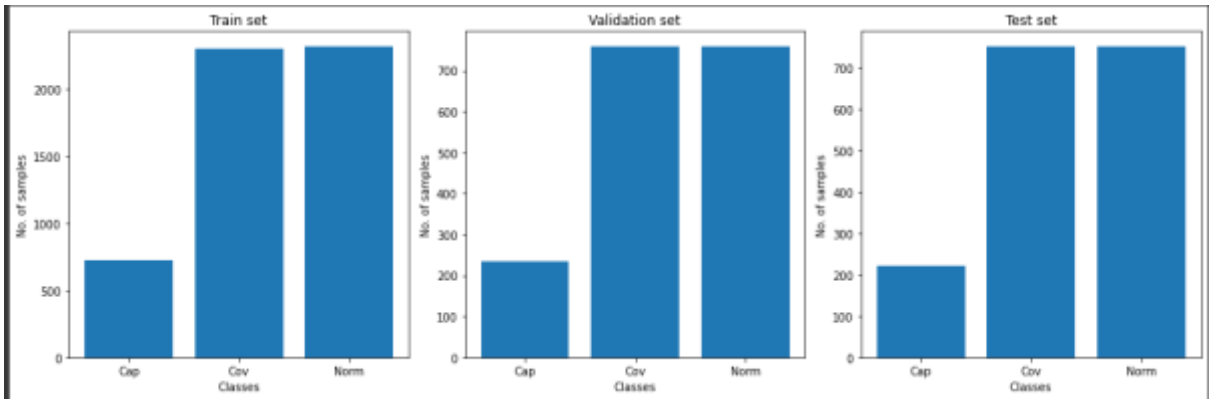
```
Found 1750 images belonging to 3 classes.
```

Extracting Test data

```
[6]  1 test_datagen = ImageDataGenerator(rescale=1./255)
     2 test_set = test_datagen.flow_from_directory(
     3         'data/Test',
     4         target_size=(256, 256),
     5         batch_size=150,
     6         class_mode='categorical',
     7         shuffle=True)
```

```
Found 1720 images belonging to 3 classes.
```

2.1. <u>Data visualisation:</u> we can see that data is highly imbalance that's why we need to apply different weights to all classes.




```
Labels:  {'Cap': 0, 'Cov': 1, 'Norm': 2}
============================================================
Instance of Train set:  5349
No. of batch:  168
Output dimensions of image:  (256, 256, 3)
Sample per class:  dict_items([(0, 724), (1, 2307), (2, 2318)])
============================================================
Instance of Validation set 1750
No. of batch:  55
Output dimensions of image:  (256, 256, 3)
Sample per class:  dict_items([(0, 234), (1, 758), (2, 758)])
============================================================
Instance of Test set 1720
No. of batch:  12
Output dimensions of image:  (256, 256, 3)
Sample per class:  dict_items([(0, 220), (1, 750), (2, 750)])
```

3. Model building:

```
base_model = tf.keras.applications.resnet50.ResNet50(include_top=False, input_shape=(256, 256, 3))
avg =  tf.keras.layers.GlobalAveragePooling2D()(base_model.output)
dropout = tf.keras.layers.Dropout(0.2)(avg)
output = tf.keras.layers.Dense(3,activation ="softmax")(dropout)
model = tf.keras.Model(inputs=base_model.input,outputs=output)
```

```
model.summary()
```

```
Total params: 23,593,859
Trainable params: 23,540,739
Non-trainable params: 53,120
```

```
from keras.callbacks import ModelCheckpoint,CSVLogger
mc = ModelCheckpoint('resnet50_proj_1.h5', monitor='val_loss', mode='min', verbose=1, save_best_only=True)
cv = keras.callbacks.CSVLogger('resnet50_proj_1.csv', separator=',', append=False)
```

```
weights = {0:3, 1:1, 2:1}
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
history = model.fit(train_set, validation_data=valid_set, class_weight=weights, callbacks=[mc, cv] , epochs=15, verbose=1)
```

4. Evaluation

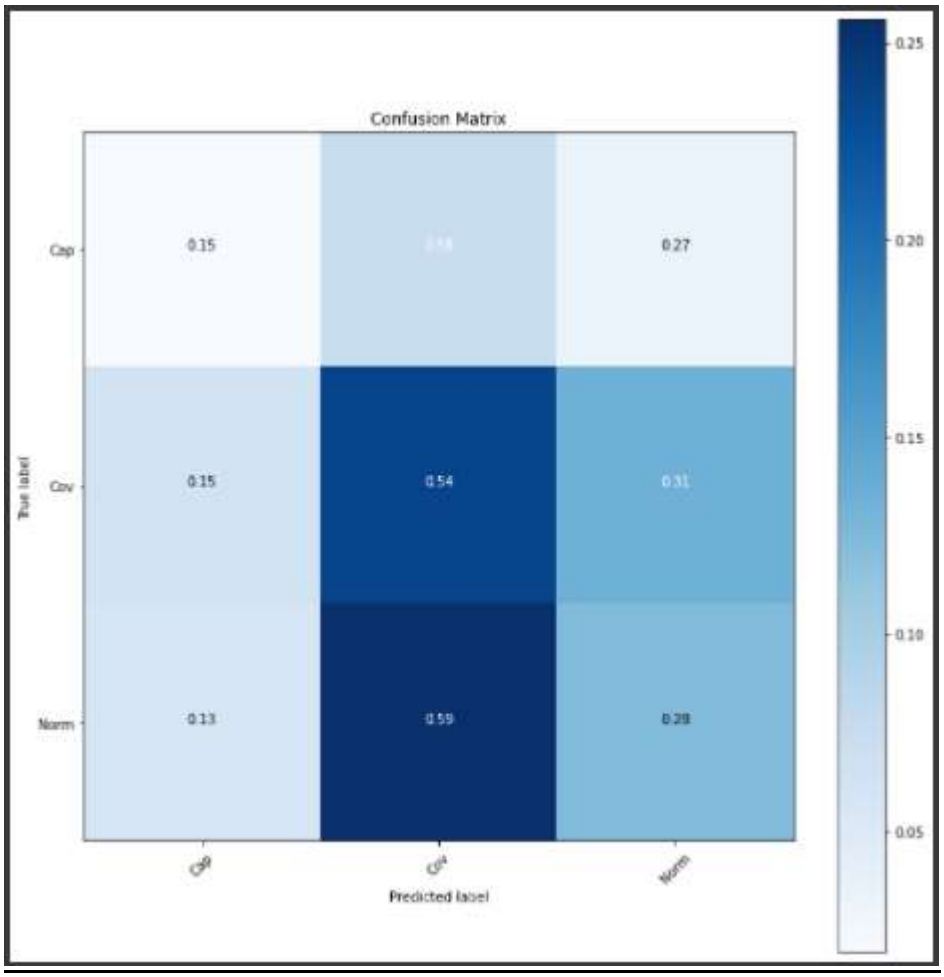
Data set	Loss	Accuracy
Train	0.1557	0.9540
Validation	2.1603	0.7229
Test	2.4848	0.5634

5. Output result:

Classification report:

	precision	recall	f1-score	support
Cap	0.14	0.15	0.14	220
Cov	0.42	0.54	0.47	750
Norm	0.42	0.28	0.34	750
accuracy			0.38	1720
macro avg	0.33	0.33	0.32	1720
weighted avg	0.38	0.38	0.37	1720

Confusion matrix:



6. Conclusion:

By observing results we can conclude that model is learning because it's training accuracy is high enough but model is over fitted as well because it's did not perform well on validation and test set.