Detection of Pneumonia Using Resnet Models

MSc Research Project
Data Analytics

Sharan Mohan
Student ID: 21112762

School of Computing
National College of Ireland

Supervisor: Qurrat Ul Ain

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Sharan Mohan
Student ID:	21112762
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Qurrat Ul Ain
Submission Due Date:	01/02/2023
Project Title:	Detection of Pneumonia Using Resnet Models
Word Count:	603
Page Count:	5

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	1st February 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Detection of Pneumonia Using Resnet Models

Sharan Mohan
21112762

1 Introduction

The Configuration manual explains the steps in the implementation process of the research Detection of Pneumonia using resnet models in detail. The required hardware and software specifications for the research is noted in configuration manual. The main objective of this research is to detect the disease Pneumonia using type of resnet models (Resnet-50,Resnet-101,Resnet-152 and Inception-Resnet) with the help of x-ray images in a less cost and time efficient way.

2 System Specification

This research Project was implemented on open source software named Jupyter Notebook.It is a mathematica notebook evolved into robust tool (Randles et al.; 2017) The required hardware and software specifications are as follows:

2.1 Hardware

- RAM: 16 GB
- GPU: Nvidia RTX 3060
- SSD: 1TB

2.2 Software

This project was executed on on Python Programming Language. From exporting libraries, data extraction, Pre-processing, Exploratory Data Analysis, Model Building and Evaluation Metrics all are executed on python in Jupytre Notebook.

3 Import Libraries

The first step of this research is to import the required libraries such as numpy and torchvision. If any other libraries are required they are imported later on the project.

```
In [2]: from __future__ import print_function, division

        import torch
        import torch.nn as nn
        import torch.optim as optim
        from torch.optim import lr_scheduler
        import numpy as np
        import torchvision
        from torchvision import datasets, models, transforms
        import matplotlib.pyplot as plt
        import time
        import os
        import copy
        from torch.utils.data.sampler import SubsetRandomSampler
        from sklearn.model_selection import train_test_split
```

Figure 1: Importing Libraries

4 Data Extraction

In this stage, the data has been imported to the notebook with the help of torchvision library as it is a image folder. And the imported dataset has been read.

```
In [2]: train_dataset = torchvision.datasets.ImageFolder(root='C:/Users/chris/OneDrive/Documents/Nci/sem 3/Research Project/Pediatric Che
        test_dataset = torchvision.datasets.ImageFolder(root='C:/Users/chris/OneDrive/Documents/Nci/sem 3/Research Project/Pediatric Ches

In [3]: train_dataset

Out[3]: Dataset ImageFolder
            Number of datapoints: 5232
            Root location: C:/Users/chris/OneDrive/Documents/Nci/sem 3/Research Project/Pediatric Chest X-ray Pneumonia/train

In [4]: test_dataset

Out[4]: Dataset ImageFolder
            Number of datapoints: 624
            Root location: C:/Users/chris/OneDrive/Documents/Nci/sem 3/Research Project/Pediatric Chest X-ray Pneumonia/test
```

Figure 2: Importing Data

5 Exploratory Data Analysis

The Exploratory Data Analysis has been carried out using the seaborn Library which has been imported. A diagrammatic representation of the class distribution plotted with a simple loop as there is only two classes to check the class imbalance.

```
In [5]: import seaborn as sns
        l = []
        for i in train_dataset:
            if(i[1] == 0):
                l.append("Normal")
            else:
                l.append("Pneumonia")
        sns.set_style('darkgrid')
        sns.countplot(l)
```

Figure 3: Class Distribution

6 Data Pre-processing

The dataset has been already splitted, so the train data is augmented using the image data generator with various augmentation techniques with the help of keras library. Some basic augmentation such as height and width change are performed.The augmentation techniques are performed only on the train data.

```
In [7]: from keras.preprocessing import image
        from keras.preprocessing.image import ImageDataGenerator
        from keras.applications.resnet_v2  import preprocess_input
        train_datagen = ImageDataGenerator(
            preprocessing_function=preprocess_input,
            rotation_range=40,
            width_shift_range=0.2,
            height_shift_range=0.2,
            shear_range=0.2,
            zoom_range=0.2,
            horizontal_flip=True,
            fill_mode='nearest')
```

Figure 4: Augmentation Techniques Implemented

7 Deep Learning Models

7.1 Model Building

There are four residual network models used in this research with the help of transfer learning. The four models are Resnet-50, Resnet-101, Resnet-152 and Inception-resnet.

The models used in this research are imported using keras library. The models is added with a flatten and dense layer at the end with weights of imagenet.

7.1.1 Resnet-50

```
In [12]: from keras.applications.resnet_v2 import ResNet50V2
         from keras.applications.resnet_v2  import preprocess_input

In [13]: IMAGE_SIZE = [224, 224]
         resnet_v2 = ResNet50V2(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)

In [14]: resnet_v2 .input

Out[14]: <KerasTensor: shape=(None, 224, 224, 3) dtype=float32 (created by layer 'input_1')>

In [15]: for layer in resnet_v2 .layers:
           layer.trainable = False

In [16]: from keras.layers import Dense, Flatten, Conv2D, MaxPooling2D
         from keras.models import Model, Sequential
         x = Flatten()(resnet_v2 .output)
         prediction = Dense(2, activation='softmax')(x)
         model = Model(inputs=resnet_v2 .input, outputs=prediction)
         model.summary()
```

Figure 5: Resnet-50

7.1.2 Resnet-101

```
In [12]: from keras.applications.resnet_v2 import ResNet101V2
         from keras.applications.resnet_v2  import preprocess_input

In [13]: IMAGE_SIZE = [224, 224]
         resnet_v2 = ResNet101V2(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)

In [14]: resnet_v2 .input

Out[14]: <KerasTensor: shape=(None, 224, 224, 3) dtype=float32 (created by layer 'input_1')>

In [15]: for layer in resnet_v2 .layers:
           layer.trainable = False

In [16]: from keras.layers import Dense, Flatten, Conv2D, MaxPooling2D
         from keras.models import Model, Sequential
         x = Flatten()(resnet_v2 .output)
         prediction = Dense(2, activation='softmax')(x)
         model = Model(inputs=resnet_v2 .input, outputs=prediction)
         model.summary()
```

Figure 6: Resnet-101

7.1.3 Resnet-152

```
In [12]: from keras.applications.resnet_v2 import ResNet152V2
         from keras.applications.resnet_v2  import preprocess_input

In [13]: IMAGE_SIZE = [224, 224]
         resnet_v2 = ResNet152V2(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)

In [14]: resnet_v2 .input

Out[14]: <KerasTensor: shape=(None, 224, 224, 3) dtype=float32 (created by layer 'input_1')>

In [15]: for layer in resnet_v2 .layers:
           layer.trainable = False

In [16]: from keras.layers import Dense, Flatten, Conv2D, MaxPooling2D
         from keras.models import Model, Sequential
         x = Flatten()(resnet_v2 .output)
         prediction = Dense(2, activation='softmax')(x)
         model = Model(inputs=resnet_v2 .input, outputs=prediction)
         model.summary()
```

Figure 7: Resnet-152

7.1.4 Inception-Resnet

```
In [11]: from keras.models import Model, Sequential
         from keras.applications.inception_resnet_v2 import InceptionResNetV2
         from keras.applications.inception_resnet_v2  import preprocess_input

In [12]: IMAGE_SIZE = [224, 224]
         inceptionresnet = InceptionResNetV2(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)

In [13]: inceptionresnet.input

Out[13]: <KerasTensor: shape=(None, 224, 224, 3) dtype=float32 (created by layer 'input_1')>

In [14]: for layer in inceptionresnet.layers:
           layer.trainable = False

In [15]: from keras.layers import Dense, Flatten, Conv2D, MaxPooling2D
         from keras.models import Model, Sequential
         x = Flatten()(inceptionresnet.output)
         prediction = Dense(2, activation='softmax')(x)
         model = Model(inputs=inceptionresnet.input, outputs=prediction)
         model.summary()
```

Figure 8: Inception-Resnet

7.2 Model Fitting

The models are fitted into the model with a normal model fit function. The optimizer and loss functions are imported and compiled used model.compile function using optimizers.

```
In [17]: from keras import optimizers


         adam = optimizers.Adam(0.0001)
         model.compile(loss='categorical_crossentropy',
                       optimizer=adam,
                       metrics=['accuracy'])

In [18]:



         results = model.fit(train_set,epochs=10,
                                       validation_data=test_set)
```

Figure 9: Model Fitting

Accuracy of train and test along with the data loss of both the datasets are printed in the model fitting step itself with the help of categorical cross-entropy.

8 Evaluation

The model built is tested using the evaluation metrics.With the help of sklearn library confusion matrix is imported and the test set is fitted on it. Basic evaluation metrics such as precision, recall and f1-score are printed using the classification report.

```
In [22]: from sklearn.metrics import classification_report, confusion_matrix
         resnet101v2model=model
         Y_pred =  resnet101v2model.predict_generator(test_set)
         y_pred = np.argmax(Y_pred, axis=1)
         print('Confusion Matrix ')
         cm = confusion_matrix(test_set.classes, y_pred)
         plot_confusion_matrix(cm, classes,False, title='Confusion Matrix model 1')
```

Figure 10: Confusion Matrix

```
In [23]: print(classification_report(test_set.classes,y_pred))

                       precision    recall  f1-score   support

                    0       0.37      0.39      0.38       234
                    1       0.62      0.59      0.61       390

             accuracy                           0.52       624
            macro avg       0.49      0.49      0.49       624
         weighted avg       0.52      0.52      0.52       624
```

Figure 11: Classification Report

References

Randles, B. M., Pasquetto, I. V., Golshan, M. S. and Borgman, C. L. (2017). Using the jupyter notebook as a tool for open science: An empirical study, *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pp. 1–2.