Configuration Manual

MSc Research Project
Data Analytics

Sanjeet Shekhar
Student ID: 23209470

School of Computing
National College of Ireland

Supervisor: Shubham Subhnil

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Sanjeet Shekhar
Student ID:	23209470
Programme:	MSc. Data Analytics **Year:** 2024
Module:	MSc. Research Project
Lecturer:	Shubham Subhnil
Submission Due Date:	12/12/2024
Project Title:	Diabetic retinopathy detection a comparison of several CNN models and optimizers
Word Count:	371 **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: ……………………………………………

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):	

Configuration Manual

Sanjeet Shekhar
Student ID: 23209470

1 Introduction

The research is about comparing CNN models to get better understanding of which model is best for performing image classification. The work compares ResNet vs MobileNet vs EffcientNet to classify the dataset containing 5 stages of Diabetic Retinopathy. The work also compares the efficiency and accuracy of the model based on the optimizer chosen. ADAM optimizer is compared to Stochastic Gradient Descent (SGD) optimizer in order to find out the best optimizer for the use case.

Following is the step-by-step process demonstrating the project setup and system requirements along with the tools and libraries that are required to run the code.

2 System Configuration

The analysis work is done on Google-based cloud platform Google Colaboratory (colab). The Pro version of Google colab comes with GPU and RAM options.
The following screenshots show the System configuration for colab used in the project.

```
[ ]   1 !nvidia-smi

Thu Dec 12 03:04:22 2024
+---------------------------------------------------------------------------------------+
| NVIDIA-SMI 535.104.05             Driver Version: 535.104.05   CUDA Version: 12.2     |
|-----------------------------------------+----------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                                         |                      |               MIG M. |
|=========================================+======================+======================|
|   0  Tesla T4                       Off | 00000000:00:04.0 Off |                    0 |
| N/A   44C    P8               9W /  70W |      3MiB / 15360MiB |      0%      Default |
|                                         |                      |                  N/A |
+-----------------------------------------+----------------------+----------------------+

+---------------------------------------------------------------------------------------+
| Processes:                                                                            |
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory |
|        ID   ID                                                             Usage      |
|=======================================================================================|
|  No running processes found                                                           |
+---------------------------------------------------------------------------------------+
```

Fig 1 Colab System Specification

3 Data Collection

The dataset used in the research work is collected from open source platform.
Dataset URL: https://www.kaggle.com/datasets/sovitrath/diabetic-retinopathy-224x224-gaussian-filtered
This zip file is then uploaded to the google drive which is accessible by google colab.

Imports and Environment Setup

```
[ ]
from google.colab import drive # Data loading from google drive
drive.mount('/content/gdrive')
!unzip -q /content/gdrive/MyDrive/dataset/retina_dataset.zip
```

```
Mounted at /content/gdrive
```

```
[ ]
import os
from torchvision.datasets import ImageFolder
data_dir = './gaussian_filtered_images/gaussian_filtered_images'
os.listdir(data_dir)
if os.path.exists('./gaussian_filtered_images/gaussian_filtered_images/export.pkl'):
  os.remove(data_dir + '/export.pkl') # removing unecessary file from dataset
print(os.listdir(data_dir))
print(ImageFolder(data_dir).root)
```

```
['Moderate', 'No_DR', 'Mild', 'Severe', 'Proliferate_DR']
./gaussian_filtered_images/gaussian_filtered_images
```

Fig 2. Google Drive to colab setup and other environment setup

4 Implementation

4.1 Libraries used to research

```
# Essential imports
import torch
import torch.nn as nn
import torch.optim as optim
from torchvision import datasets, transforms, models
from torch.utils.data import DataLoader
import time
import cv2
from torchvision.transforms import ToPILImage, ToTensor, Compose, RandomRotation, Lambda
from torch.optim.lr_scheduler import StepLR
from torchvision import datasets, transforms
from torch.utils.data import DataLoader
```

Fig 3. Libraries used for analysis

The figure shows the libraries used for analysis.
Some of the major libraries include

- Torch
- Torchvision
- Cv2

```
# Dataset preparation
import os
import random
import shutil
from torchvision import datasets, transforms
from torch.utils.data import DataLoader, random_split

```

Fig 4. Libraries used to set colab and perform other preprocessing steps

```
# Visualization setup
from torchvision.utils import make_grid
import matplotlib.pyplot as plt
import numpy as np
from scipy.interpolate import make_interp_spline

```

Fig 5. Libraries used to perform visualization before and after analysis

4.2 Dataset

The dataset zip file contains 5 directories that contain images of eyes affected by various stages of diabetic retinopathy.

- Mild
- Moderate
- No_DR
- Proliferate_DR
- Severe
- export.pkl (pkl file that needs to be dropped before proceeding)
-

```
[2]
import os
from torchvision.datasets import ImageFolder
data_dir = './gaussian_filtered_images/gaussian_filtered_images'
os.listdir(data_dir)
if os.path.exists('./gaussian_filtered_images/gaussian_filtered_images/export.pkl'):
    os.remove(data_dir + '/export.pkl') # removing unecessary file from dataset
print(os.listdir(data_dir))
print(ImageFolder(data_dir).root)

['Moderate', 'No_DR', 'Mild', 'Severe', 'Proliferate_DR']
./gaussian_filtered_images/gaussian_filtered_images
```

Fig 6. Directories setup and cleanup of unnecessary files

4.3 The Flow of the Implementation

- Setup directories for image classification
- Visualize the sample images
- Define model architectures
- Setup training and evaluations
- Run and compare models
- Choose best model

- Run training again on same model to compare optimizers

```python
# Function to visualize dataset samples
def show_images_from_dataset(dataset, n_images=8):
    loader = torch.utils.data.DataLoader(dataset, batch_size=n_images, shuffle=True)
    data_iter = iter(loader)
    images, labels = next(data_iter)

    grid_img = make_grid(images, nrow=4, normalize=True)
    plt.figure(figsize=(10, 5))
    plt.imshow(grid_img.permute(1, 2, 0))
    plt.title("Sample Images from Dataset")
    plt.axis('off')
    plt.show()

# Visualize a few samples from the train dataset
train_dataset = datasets.ImageFolder(train_path, transform=transforms.ToTensor())
show_images_from_dataset(train_dataset)

```

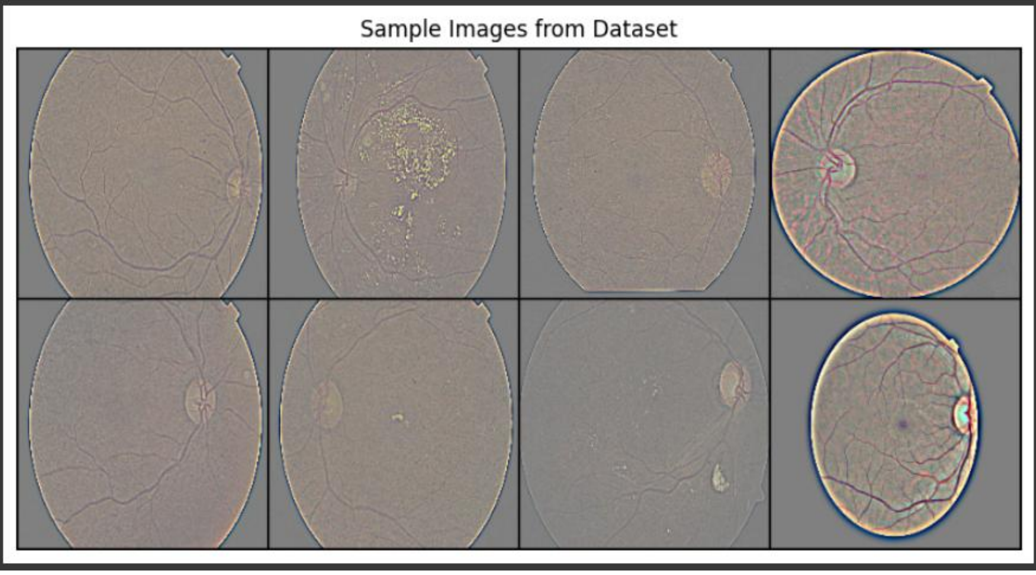

Fig 7. Samples for the loaded and cleaned dataset

```python
    plt.text(epochs[-1], train_accuracy[-1], f'{train_accuracy[-1]:.2f}', fontsize=12, color='teal', ha='center')
    plt.text(epochs[-1], train_loss[-1], f'{train_loss[-1]:.2f}', fontsize=12, color='orange', ha='center')

    # Styling
    plt.title(f'Model Accuracy vs Loss Visualization for {model_name}' , fontsize=18, fontweight='bold', color='darkblue')
    plt.xlabel('Epochs', fontsize=14)
    plt.ylabel('Accuracy', fontsize=14)
    plt.xticks(fontsize=12)
    plt.yticks(fontsize=12)
    plt.grid(alpha=0.4, linestyle='--')
    plt.legend(fontsize=14, loc='upper right')
    plt.tight_layout()
    plt.show()
```

```
Training model: resnet
/usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:208: UserWarning: The parameter 'pretrained' is deprecated since 0.13 and may be removed in the future, please use 'weights' instead.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/torchvision/models/_utils.py:223: UserWarning: Arguments other than a weight enum or `None` for 'weights' are deprecated since 0.13 and may be removed in the future. The
  warnings.warn(msg)
Downloading: "https://download.pytorch.org/models/resnet18-f37072fd.pth" to /root/.cache/torch/hub/checkpoints/resnet18-f37072fd.pth
100%|██████████| 44.7M/44.7M [00:00<00:00, 215MB/s]
Epoch 0/199
----------
train Loss: 1.4650 Acc: 0.3797
val Loss: 1.3586 Acc: 0.4529
Epoch 1/199
----------
train Loss: 1.2680 Acc: 0.5159
val Loss: 1.3046 Acc: 0.5102
Epoch 2/199
----------
train Loss: 1.2166 Acc: 0.5500
val Loss: 1.2421 Acc: 0.5375
Epoch 3/199
----------
train Loss: 1.1677 Acc: 0.5708
val Loss: 1.1976 Acc: 0.5334
Epoch 4/199
----------
train Loss: 1.1490 Acc: 0.5924
val Loss: 1.1556 Acc: 0.5553
```

Fig 8. Running models to pick the best model.

5 Evaluating model performance

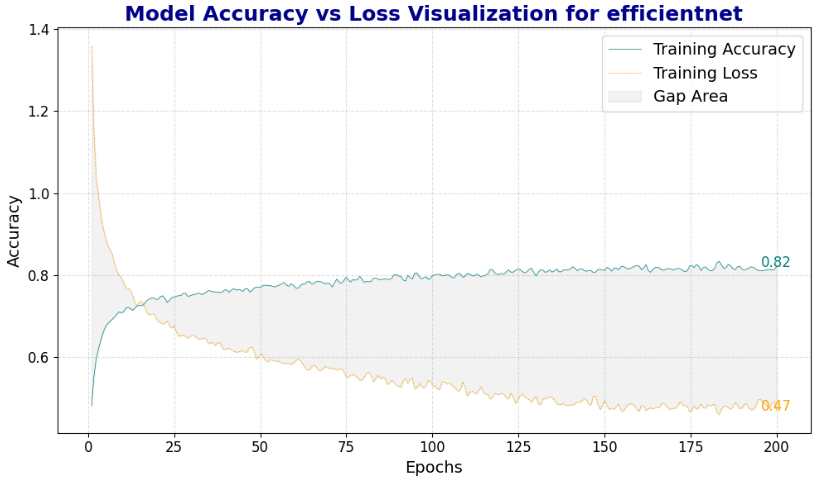

Fig 9. EfficientNet Performance

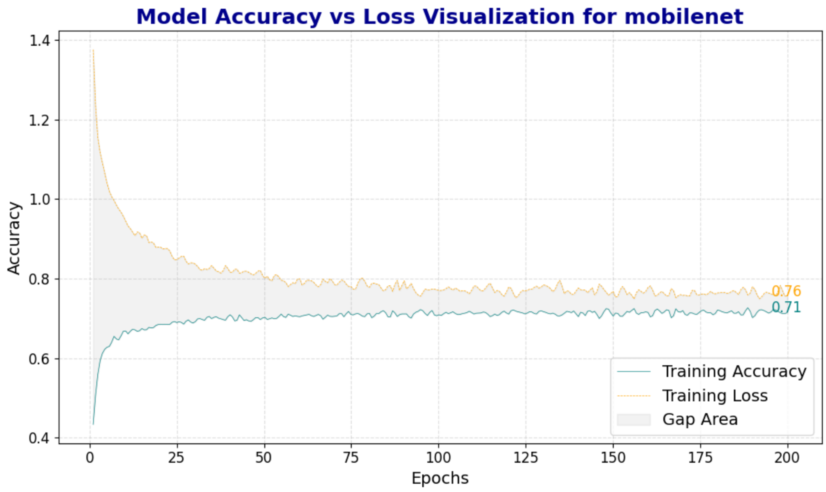

Fig 10. MobileNet Performance

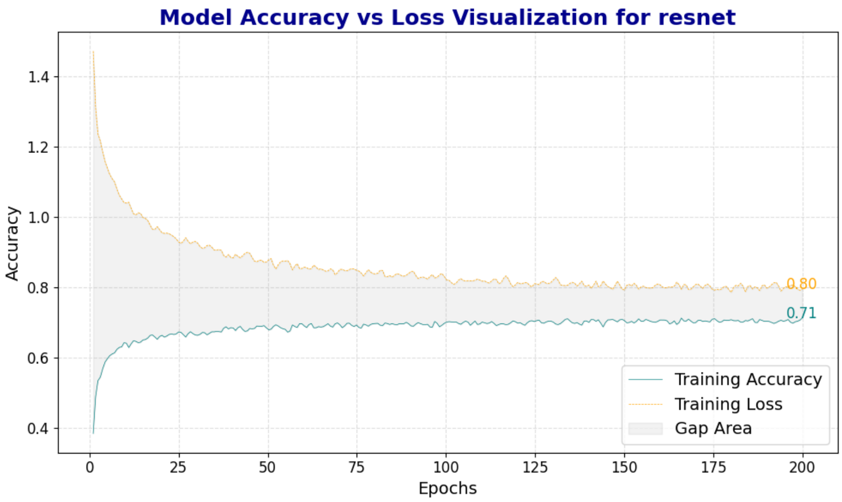

Fig 11. ResNet Performance