Configuration Manual

MSc Research Project
Cybersecurity

Mahavir Gala
Student ID: 22208208

School of Computing
National College of Ireland

Supervisor: Joel Aleburu

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Mahavir Gala		
Student ID:	22208208		
Programme:	Masters in Cybersecurity	**Year:**	2023-2024
Module:	MSc Research Project		
Supervisor:	Joel Aleburu		
Submission Due Date:	12/08/2024		
Project Title:	Adult content filtering using Machine learning		
Word Count:	497	**Page Count:**	9

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.
<u>ALL</u> 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: Mahavir Gala

Date: 12/08/2024

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

Mahavir Gala
22208208

1 Introduction

The document outlines the steps required to successfully execute the project. The project has been developed on an online cloud platform for executing "python" language codes, with the help of necessary python packages and libraries.

2 Experimental Setup

2.1 System Configuration

Hardware Used in this Experiment	Version	Purpose
Acer Aspire A315-41 Processor: AMD64 Family 23 Model 17 Stepping 0 AuthenticAMD ~2000 Mhz Total Physical Memory: 3,485 MB Available Physical Memory: 240 MB Virtual Memory: Max Size: 10,831 MB Virtual Memory: Available: 4,022 MB	OS Name: Microsoft Windows 10 Home Single Language OS Version: 10.0.19045 N/A Build 19045	Workstation

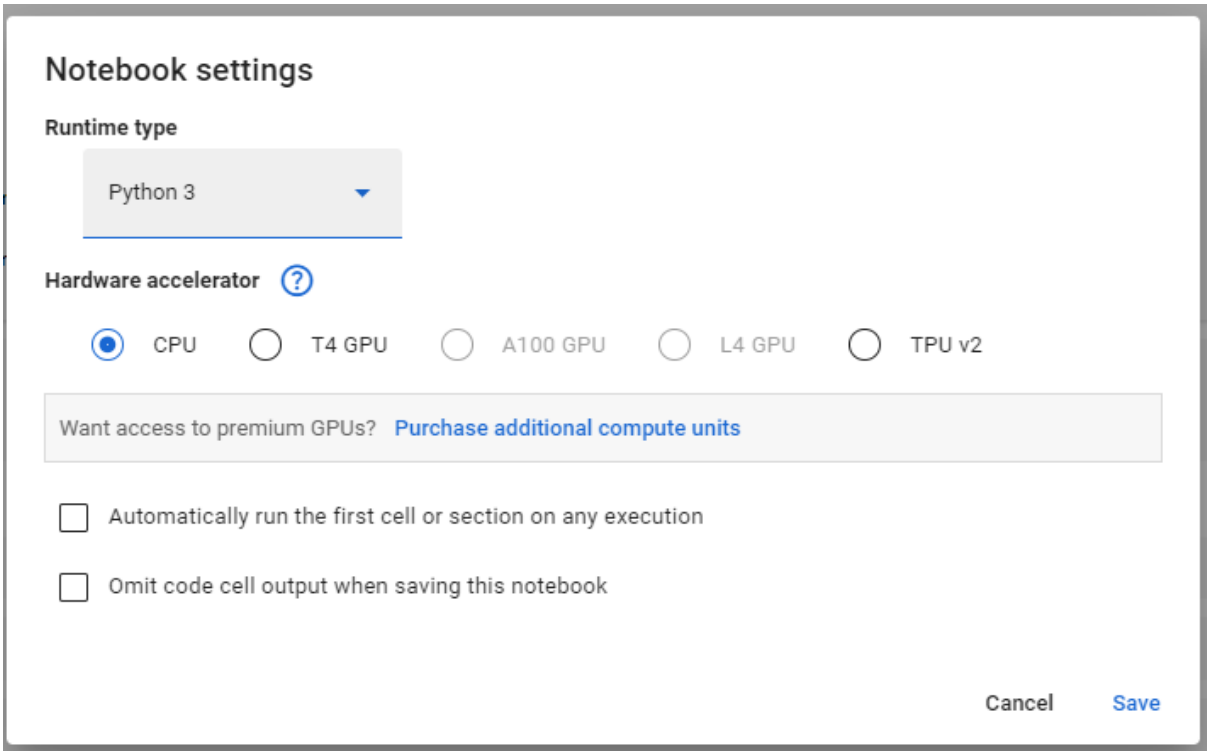
2.2 Software Used in this Experiment

Google colab has been used due to its easy accessibility, pre-installed packages and the ease of running python codes. Since Machine learning algorithms have been developed, a number of libraries were imported such as:

Packages

- Random (Built-in)
- Shutil (Built-in)
- Numpy 1.26.4
- cv2 4.10.0
- tensorflow 2.17.0
- sklearn 1.3.2
- Keras 2.12.0

2.2.1 The developer needs a Google account and a browser to access the online "Jupyter" notebook environment. Running the code requires opening the notebook and clicking on the "run" or "play" button associated with each cells.



2.3 Dataset

The dataset contained 2 folders, one containing normal images and the other having "adult" or "conspicuous" images. More than 7000 images were collected in the dataset.

2.3.1 Steps to upload dataset into Google Collab

Uploading the image folder file to Google Drive

2) Mount Google Drive in Google Colab

```
[ ] drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
[ ] os.chdir('/content/drive/My Drive/')

    with zipfile.ZipFile('image dataset.zip', 'r') as zip_ref:
        zip_ref.extractall('image_dataset')
```

3) Extracting the images

```
[4] def extract_images(src_dir, dest_dir, num_images):
        if not os.path.exists(dest_dir):
            os.makedirs(dest_dir)

        files = os.listdir(src_dir)
        for file in files[:num_images]:
            shutil.copy(os.path.join(src_dir, file), dest_dir)

    # Define directories
    train_dir = '/content/drive/MyDrive/image_dataset/image_dataset/image dataset/train'
    test_dir = '/content/drive/MyDrive/image_dataset/image_dataset/image dataset/train'
```

```
[5] # Create new directories for the extracted images
    new_train_dir = 'extracted_images/train'
    new_val_dir = 'extracted_images/val'
    new_test_dir = 'extracted_images/test'
```

```
[6] # Extract images
    extract_images(os.path.join(train_dir, '1'), os.path.join(new_train_dir, '1'), 1500)
    extract_images(os.path.join(train_dir, '2'), os.path.join(new_train_dir, '2'), 1500)
    extract_images(os.path.join(test_dir, '1'), os.path.join(new_val_dir, '1'), 500)
    extract_images(os.path.join(test_dir, '2'), os.path.join(new_val_dir, '2'), 500)
    extract_images(os.path.join(test_dir, '1'), os.path.join(new_test_dir, '1'), 250)
    extract_images(os.path.join(test_dir, '2'), os.path.join(new_test_dir, '2'), 250)
```

3 Implementation Steps

3.1 Importing necessary libraries

```
import random
import shutil
import cv2
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import sklearn
from sklearn.model_selection import train_test_split
import os
from google.colab import drive
import zipfile
from tensorflow.keras.applications import VGG16
from tensorflow.keras.models import Model
from keras.optimizers import Adam
from tensorflow.keras.layers import Dense, Flatten, Dropout
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, confusion_matrix
import numpy as np
```

3.2 Data Preparation

The experiment involves preparing a dataset of images by extracting and organizing them into training, validation, and test sets. This is done to ensure the model has sufficient and well-organized data to learn from and evaluate its performance.

3.3 Data Augmentation

Data augmentation techniques (e.g., shear, zoom, and horizontal flip) are applied to the training data to improve the model's robustness and generalization capabilities by artificially increasing the diversity of the training data.

```
[ ]  # Data Augmentation for Training
     train_datagen = ImageDataGenerator(
         rescale=1./255,
         shear_range=0.2,
         zoom_range=0.2,
         horizontal_flip=True
     )
```

3.4 Model Building

A Convolutional Neural Network (CNN) is built using the VGG16 architecture, which is a well-established pre-trained model. The VGG16 model is used as a base for feature extraction, with additional custom layers added to adapt it for the specific classification task.

```
# Load the VGG16 model
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5
58889256/58889256 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
```

3.5 Model Training

The model is trained on the prepared training data and evaluated on the validation set. The goal is to fine-tune the model to accurately classify images into the two categories while avoiding overfitting.

3.6 Model Evaluation

The model's performance is assessed using the test set. Metrics such as accuracy, loss, classification report, and confusion matrix are calculated to evaluate how well the model performs in classifying images.

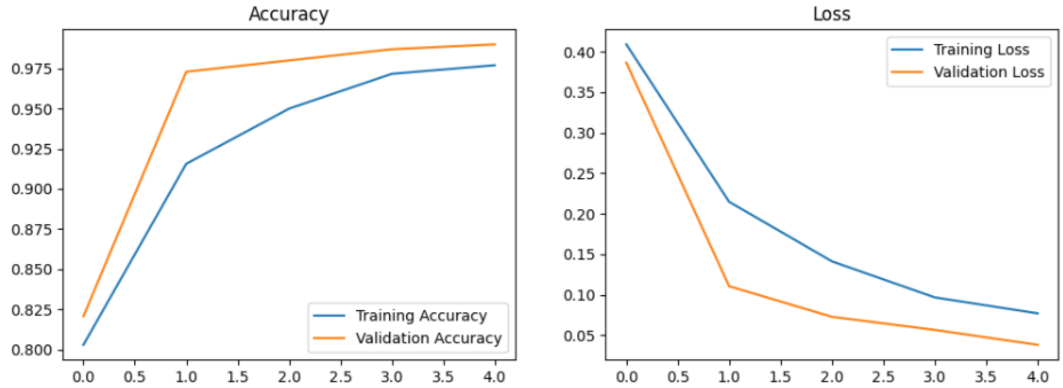
```
Epoch 1/5
94/94 ━━━━━━━━━━━━━━━━━━━━ 2427s 26s/step - accuracy: 0.7099 - loss: 0.5453 - val_accuracy: 0.8206 - val_loss: 0.3863
Epoch 2/5
94/94 ━━━━━━━━━━━━━━━━━━━━ 2416s 25s/step - accuracy: 0.8949 - loss: 0.2586 - val_accuracy: 0.9729 - val_loss: 0.1106
Epoch 3/5
94/94 ━━━━━━━━━━━━━━━━━━━━ 2359s 25s/step - accuracy: 0.9524 - loss: 0.1443 - val_accuracy: 0.9800 - val_loss: 0.0729
Epoch 4/5
94/94 ━━━━━━━━━━━━━━━━━━━━ 2354s 25s/step - accuracy: 0.9726 - loss: 0.0983 - val_accuracy: 0.9870 - val_loss: 0.0568
Epoch 5/5
94/94 ━━━━━━━━━━━━━━━━━━━━ 2348s 25s/step - accuracy: 0.9690 - loss: 0.0897 - val_accuracy: 0.9900 - val_loss: 0.0384
```

```
[ ] # Evaluate the model
    test_loss, test_accuracy = model.evaluate(test_generator)
    print(f'Test Accuracy: {test_accuracy}')
    print(f'Test Loss: {test_loss}')

16/16 ━━━━━━━━━━━━━━━━━━━━ 286s 18s/step - accuracy: 0.9905 - loss: 0.0454
Test Accuracy: 0.9879759550094604
Test Loss: 0.04213355481624603
```

3.7 Visualization

Accuracy and loss curves are plotted to visualize the training process and to understand how the model's performance evolves over epochs.


Accuracy
Training Accuracy
Validation Accuracy
Loss
Training Loss
Validation Loss