National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Mohammad Aman
Student ID: X21109079

School of Computing
National College of Ireland

Supervisor: Paul Stynes

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Mohammad Aman
Student ID:	X21109079
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Paul Stynes
Submission Due Date:	15/12/2022
Project Title:	Configuration Manual
Word Count:	XXX
Page Count:	6

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:	15th December 2022

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

Mohammad Aman
X21109079

1 Introduction

The goal of this document is to give all the directions required to replicate the deep learning framework for memory retrieval from lifelogging data.

2 Hardware Requirements

The project was implemented under the following configurations:

2.1 Local Machine

64 Bit operating system, Windows 11 with 11th Gen Intel(R) Core(TM) i5-1155G7 @ 2.50GHz and 8 GB RAM.

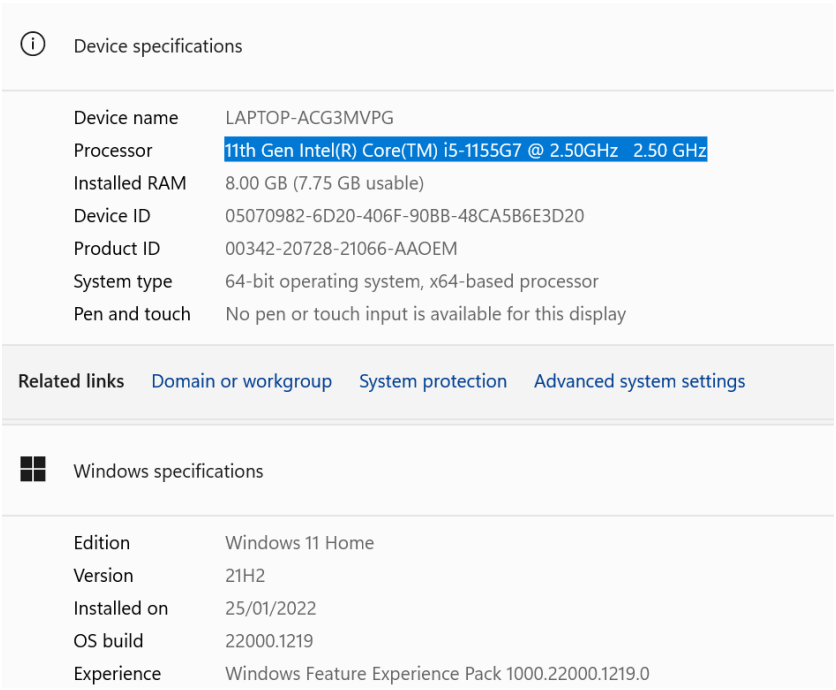

Figure 1: Hardware Configuration on Local Machine.

2.2 GPU Configuration

The major part of the project implementation was done on AWS EC2 remote server. The server g4ad.4xlarge was used with Ubuntu platform. The considered lifelogging dataset EgoRoutine was approximately of size 57 Gb. The new dataset with all the files merged was also replicated for retrieval. Image embedding were also stored on the server. The cloud storage of 200 GB was taken with the server. The screenshot of the configuration could not be incorporated as the server is now stopped automatically by NCI.

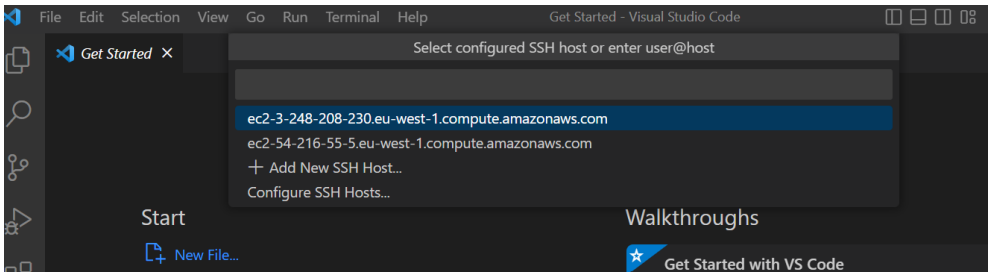

Figure 2: AWS Server

3 Software Requirement

The project is implemented using the programming language "Python". Initially, to derive embeddings, the data was migrated on the server using Filezilla as shown in Fig. 3.

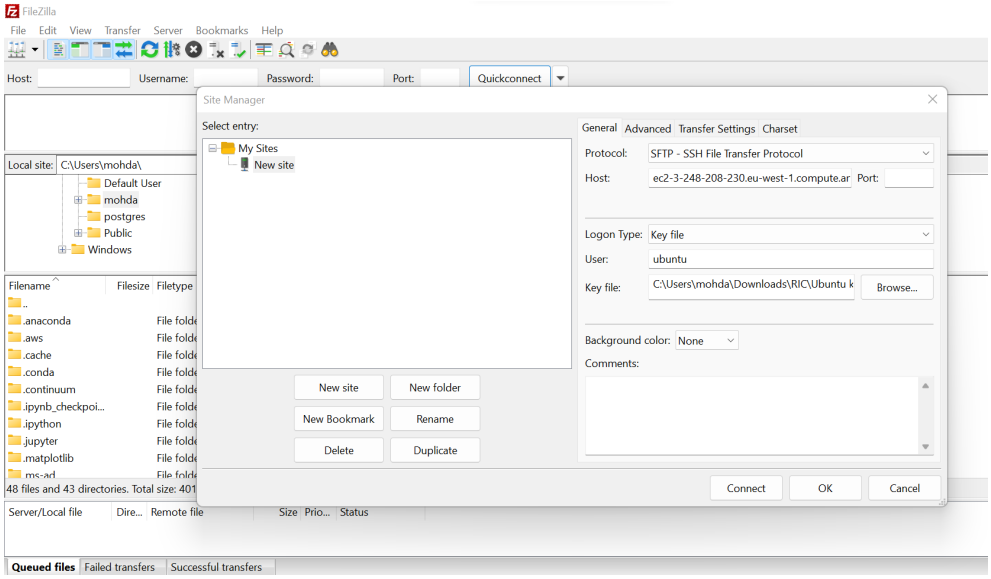

Figure 3: Filezilla for data migration

The software Putty was used to login into Ubuntu server and install python libraries, creating virtual environments and screens as shown in Fig. 4.

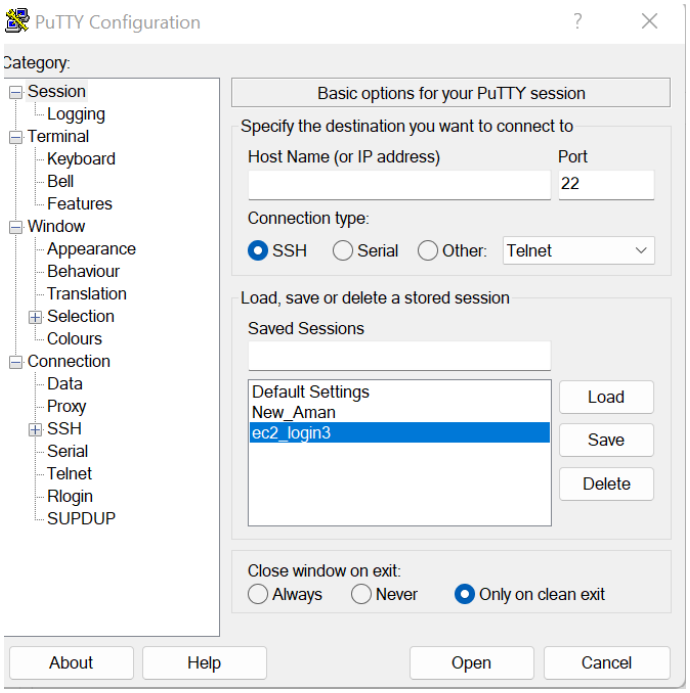

Figure 4: Putty Terminal.

The server was connected remotely through SSH in Visual Studio Code. Conda was used with Python to implement the code in VS Code as shown in 5.

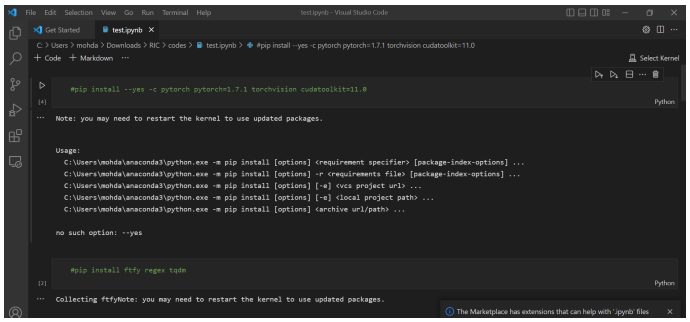

Figure 5: Visual Studio Code Interface.

The retrieval framework was implemented on the local host in Jupyter notebook using Anaconda Navigator. The navigator can be used to open Jupyter notebook and run python code and retrieve images.

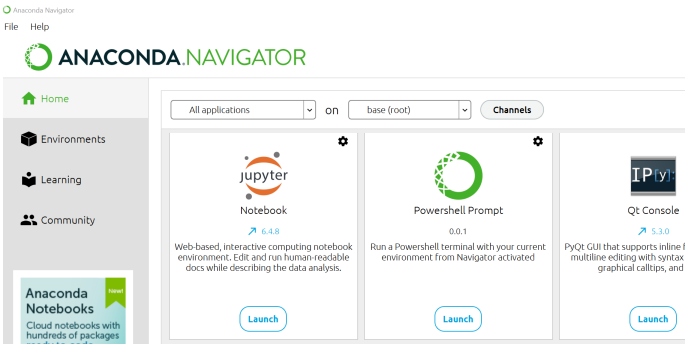

Figure 6: Anaconda Navigator.

4 Package Requirements

The python packages were installed using pip and conda command in jupyter notebook and Visual Studio Code respectively. The packages installed are listed below:

- pandas
- numpy
- clip
- torch
- pathlib
- Ipython

The clip libraries can be installed from github using the following link:`https://github.com/openai/CLIP`

5 Dataset Description

The dataset is called EgoRoutine which is created using a wearable camera by seven users over 104 days. The dataset size is 57.3 gigabytes that contain more than 100000 images in multiple folders. The dataset is publicly available for research purposes at:
`http://www.ub.edu/cvub/dataset/egoroutine/`

6 Model Preparation

The test.ipynb file available in the artefacts zip file contains the code implementation to install libraries, loading models and deriving image embedding numpy files. The code also contains implementation to save the features (numpy files) and saving image indexes in a separate file.

6.1 CLIP Models (ResNet50x64, ViT-L/14, ViT-L/14@336px) Implementation

- The dataset is read through subfolders.
- The CLIP models are loaded individually.
- The data is passed through the image encoders of the model and embeddings are derived that contains image semantics.
- Images are ranked based on the cosine similarity with the textual query.
- Final result is retrieved in the jupyter notebook (see Fig. 7 and 8).

```
In [130]: # Load the open CLIP model
          device = "cuda" if torch.cuda.is_available() else "cpu"
          model_rn, preprocess_rn = clip.load("ViT-L/14", device=device)

          model_vit14, preprocess_vit14 = clip.load("ViT-L/14@336px", device=device)

In [131]: features_path_rn = Path("C:/Users/mohda/Downloads/RIC/output/output/features/")
          features_path_vit14 = Path("C:/Users/mohda/Downloads/RIC/output/output/features_ViT-L/14336px/")


          # root folder path
          lifelog_dataset_path = Path("C:/Users/mohda/Downloads/RIC/Merged_Data/")

In [132]: # Load the features and the corresponding IDs of both the models
          photo_features_rn = np.load(features_path_rn / "features.npy")
          photo_ids_rn = pd.read_csv(features_path_rn / "photo_ids.csv")

In [133]: photo_features_vit14 = np.load(features_path_vit14 / "features.npy")
          photo_ids_vit14 = pd.read_csv(features_path_vit14 / "photo_ids.csv")

In [175]: #search_query = "I was cooking dinner. There was a microwave in the kitchen"
          #search_query = "I went shopping to buy clothes"
          #search_query = "Went on a trip. A guide was addressing people"
          #search_query = "I was sitting in a room. Whatsapp web was open on laptop"
          search_query = "Went on a group tour. A guide was addressing the crowd"
          #search_query= "There was car racing on TV. It was Formula 1 motor racing"
```

Figure 7: CLIP Implementation 1.

```
# Get all metadata for this photo
photo_data.append(photo_id)
#photo_data = photos[photos["photo_id"] == photo_id].iloc[0]
#url=date[0:4]+'-'+date[4:6]+'-'+date[6:]+ '/' + photo_id

# Display the photo
display(Image(filename='C:/Users/mohda/Downloads/RIC/Merged_Data/'+photo_id+'.jpg',width = 400,height=300))
```

Figure 8: CLIP Implementation 2.

The same procedure is followed to apply ensemble methods. The ensemble.ipynb file is provided in the code artifact zip file.

7 Steps to Implement Deep Learning Memory Retrieval Framework

- Login and setup server in FileZilla to transfer data to server.
- Login and setup server in Putty terminal to install packages, create environments.

- Login and setup server in Visual Studio Code to run python code using conda. The screen should be created in the environment soo that the code can keep running even if the local system is shutdown.
- The data is further migrated to local computer using derived embeddings files.
- The code is run in jupyter notebook using Anaconda Navigator.
- The natural language query is given in the notebook cell, for example, I was using laptop in a bedroom. The results are retrieved in the notebook and true images are identified from all the relevant images (see Fig. 7 and 8 for search query and images retrieval).