National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Rohit Pimpale
Student ID: X23268620

School of Computing
National College of Ireland

Supervisor: Rejwanul Haque

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Rohit Pimpale
Student ID:	X23268620
Programme:	Data Analytics
Year:	2025
Module:	MSc Research Project
Supervisor:	Rejwanul Haque
Submission Due Date:	11/08/2025
Project Title:	Configuration Manual
Word Count:	1203
Page Count:	10

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:	10th August 2025

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

Rohit Pimpale
X23268620

1. Hardware Specifications

- **Processor:** Intel Core i7-1260P (12 cores, base clock 2.10 GHz)
- **Graphics:** Integrated Intel Iris Xe Graphics
- **RAM:** 16 GB DDR4
- **Storage:** 1 TB NVMe SSD — sufficient to handle the *UTKFace* dataset (24,000 images), multiple saved model weights, Grad-CAM outputs, and training logs
- **Operating System:** Windows 11 Pro (64-bit)

2. Python Environment

2.1 Environment Details

- **Distribution:** Anaconda Individual Edition
- **Environment Name:** python10
- **Python Version:** 3.10.12
- **IDE:** Visual Studio Code
- **Notebook Interface:** *Jupyter* Notebook

2.2 Environment Setup and Launch

To replicate this project on a different machine, you need to set up a fresh Anaconda environment that exactly matches the original one used during model development.

Step-by-Step: Creating the Environment

1. **Open Anaconda Navigator**
2. **Navigate to the "Environments" tab** on the left panel
3. **Click "Create"** (bottom left) and enter the following:
    - **Name:** python10 (*or any name you prefer*)
    - **Python Version:** 3.10.16
4. Once created, click the **Play Button** next to the environment

2.3 Package Installation

Packages were installed into this environment using `pip` or `conda`. Example:

```
pip install tensorflow==2.18.0
```

All key packages and versions used are listed in Section 3.

3 Installed Libraries

3.1 Core Dependencies

The following Python libraries were essential for implementing and running the project:

- `tensorflow==2.16.0`
  Deep learning framework used for building and training neural networks (with `Keras` backend)
- `keras==3.8.0`
  High-level API for model definition, compilation, and training
- `opencv-python==4.11.0.86`
  Image processing library used for webcam feed, image loading, resizing, and manipulation
- `numpy==2.02.2`
  Numerical computing package for array operations and data formatting
- `pandas==2.2.3`
  Data analysis and manipulation library used for tabular operations and metadata handling
- `matplotlib==3.10.0`
  Used to plot training/validation curves and Grad-CAM visualizations
- `seaborn==0.13.2`
  Statistical data visualization used for classification accuracy plots and confusion matrices
- `scikit-learn==1.6.1`
  Provided metrics like accuracy, MAE, confusion matrix; also used for data splitting and label encoding
- `tqdm==4.67.1`
  Added real-time training progress bars for batch-level visibility

3.2 Full Package List (via pip freeze)

The full list of installed packages (and their exact versions) was generated using:

```
pip freeze > requirements.txt
```

Package	Version
absl-py	2.1.0
ace_tools	0.0
aioice	0.10.1
aiortc	1.13.0
altair	5.5.0
annotated-types	0.7.0
anyio	4.9.0
arrow	1.3.0
asttokens	3.0.0
astunparse	1.6.3
attrs	25.3.0
av	14.4.0
blinker	1.9.0
cachetools	6.1.0
certifi	2024.12.14
cffi	1.17.1
charset-normalizer	3.4.1
click	8.2.1
codecarbon	3.0.2
colorama	0.4.6
comm	0.2.2
contourpy	1.3.1
cryptography	45.0.4
cycler	0.12.1
debugpy	1.8.13
decorator	5.2.1
dnspython	2.7.0
exceptiongroup	1.2.2
executing	2.1.0
fief-client	0.20.0
filelock	3.13.1
findspark	2.0.1
flatbuffers	24.12.23
fonttools	4.55.3
fsspec	2024.6.1
gast	0.6.0
gitdb	4.0.12
GitPython	3.1.44
google-crc32c	1.7.1
google-pasta	0.2.0

grpcio	1.69.0
h11	0.16.0
h5py	3.12.1
httpcore	1.0.9
httpx	0.27.2
idna	3.10
ifaddr	0.2.0
imbalanced-learn	0.13.0
imblearn	0.0
importlib_metadata	8.6.1
ipykernel	6.29.5
ipython	8.34.0
jedi	0.19.2
Jinja2	3.1.6
joblib	1.4.2
jsonschema	4.24.0
jsonschema-specifications	2025.4.1
jupyter_client	8.6.3
jupyter_core	5.7.2
jwcrypto	1.5.6
keras	3.8.0
kiwisolver	1.4.8
libclang	18.1.1
Markdown	3.7
markdown-it-py	3.0.0
MarkupSafe	3.0.2
matplotlib	3.10.0
matplotlib-inline	0.1.7
mdurl	0.1.2
ml-dtypes	0.4.1
mpmath	1.3.0
namex	0.0.8
narwhals	1.45.0
nest_asyncio	1.6.0
networkx	3.3
numpy	2.0.2
nvidia-ml-py	12.575.51
opencv-python	4.11.0.86
opencv-python-headless	4.11.0.86
opt_einsum	3.4.0
optree	0.14.0
packaging	24.2
pandas	2.2.3
parso	0.8.4
pickleshare	0.7.5
pillow	11.1.0

pip	25.1.1
platformdirs	4.3.7
prometheus$_{c}$*lient*	0.22.1
prompt_toolkit	3.0.50
protobuf	5.29.3
psutil	7.0.0
pure_eval	0.2.3
py-cpuinfo	9.0.0
py4j	0.10.9.7
pyarrow	20.0.0
pycparser	2.22
pydantic	2.11.7
pydantic_core	2.33.2
pydeck	0.9.1
pydot	4.0.1
pyee	13.0.0
Pygments	2.19.1
pylibsrtp	0.12.0
pymongo	4.12.0
pynvml	12.0.0
pyOpenSSL	25.1.0
pyparsing	3.2.1
pyspark	3.5.5
python-dateutil	2.9.0.post0
pytorch-tabnet	4.1.0
pytz	2025.2
pywin32	307
pyzmq	26.3.0
questionary	2.1.0
RapidFuzz	3.13.0
referencing	0.36.2
requests	2.32.3
rich	13.9.4
rpds-py	0.26.0
scikit-learn	1.6.1
scipy	1.15.2
seaborn	0.13.2
setuptools	75.1.0
shellingham	1.5.4
six	1.17.0
sklearn-compat	0.1.3
smmap	5.0.2
sniffio	1.3.1
stack_data	0.6.3
streamlit	1.46.1
streamlit-webrtc	0.63.3

sympy	1.13.3
tenacity	9.1.2
tensorboard	2.18.0
tensorboard-data-server	0.7.2
tensorflow	2.18.0
tensorflow_intel	2.18.0
tensorflow-io-gcs-filesystem	0.31.0
termcolor	2.3.0
threadpoolctl	3.6.0
toml	0.10.2
torch	2.7.1+cpu
torchaudio	2.7.1+cpu
torchvision	0.22.1+cpu
tornado	6.4.2
tqdm	4.67.1
traitlets	5.14.3
typer	0.16.0
types-python-dateutil	2.9.0.20250516
typing_extensions	4.12.2
typing-inspection	0.4.1
tzdata	2025.2
urllib3	2.3.0
watchdog	6.0.0
wcwidth	0.2.13
Werkzeug	3.1.3
wheel	0.44.0
wrapt	1.17.2
yaspin	3.1.0
zipp	3.21.0

4. Dataset Configuration

4.1 Dataset Overview

- **Dataset Name:** UTKFace
- **Total Images:** ˜24,000
- **Image Format:** .jpg
- **Image Type:** RGB, mostly 200×200 pixels
- **Label Format (from file name):**
  `<age>_<gender>_<ethnicity>_<date&time>.jpg`

Example:

`23_0_1_20170116223627486.jpg.chip.jpg`

- **Age:** 23
- **Gender:** 0 (Male)
- **Ethnicity:** 1 (Black)

4.3 Data Preprocessing

All images were resized and normalized beforehand to make sure they would work smoothly with various CNN backbone architectures.

Model	Input Size Used
MobileNetV2	$224 \times 224 \times 3$
ResNet50	$224 \times 224 \times 3$
EfficientNetB3	$300 \times 300 \times 3$

4.4 Data Splitting

- **Training Set:** 80%
- **Validation Set:** 20%

5. Saved Models and Files

5.1 Saved Model Files

- **.h5 files:**
  Store the complete model structure along with the trained weights.

File Name	Description
`mobilenetv2_model.h5`	Final trained MobileNetV2 model
`resnet50_model.h5`	Final trained ResNet50 model
`efficientnetb3_model.h5`	Final trained EfficientNetB3 model

Example of loading a model:

```
from tensorflow.keras.models import load_model

model = load_model('mobilenetv2_model.h5')
```

5.2 Additional Files

- **Live prediction script:**
  `live_predict.py`, a standalone Python file which performs real-time validation.

6. Steps for Running the Code

1. **Open Anaconda Navigator:**
   - Go to the `"Environments"` tab.
   - Select the environment named `python10` (or create it as described in Section 2).
2. **Install Required Libraries:**
   - In the opened terminal or notebook cell, run:

     ```
     pip install -r requirements.txt
     ```

3. **Launch the Jupyter Notebook:**
   - Open `x23268620.ipynb`
4. **Load and Preprocess the Dataset:**
   - The `UTKFace` dataset is parsed using filenames to extract age, gender, and ethnicity.
   - Images are resized to:
     - 224×224 for MobileNetV2 and ResNet50
     - 300×300 for EfficientNetB3
   - Data is split into training (80%) and validation (20%).
5. **Apply Real-Time Data Augmentation:**
   - Augmentations include:
     - Rotation, zoom, shift, brightness variation, shear, and horizontal flip
   - These augmentations are applied only on training data.
6. **Choose and Build a Model Architecture:**
   - Select one from:
     - MobileNetV2
     - ResNet50
     - EfficientNetB3
   - Load the pre-trained base (ImageNet) and add:
     - Output head for age (regression)
     - Output head for gender (binary classification)
     - Output head for ethnicity (multi-class classification)
7. **Compile the Model:**
   - Use the Adam optimizer.
   - Define the multi-output loss functions:

  - Age: MAE
  - Gender: Categorical Crossentropy
  - Ethnicity: Categorical Crossentropy
- Apply task-specific loss weights to balance learning.

8. **Train the Model:**
   - Run for 30–40 epochs using batch size 32.
   - Callbacks:
     - `EarlyStopping` (patience = 7)
     - `ReduceLROnPlateau` (patience = 3)

9. **Evaluate the Model:**
   - Print the following metrics:
     - Age prediction MAE
     - Gender classification accuracy
     - Ethnicity classification accuracy
   - Plot training vs. validation loss and accuracy curves for each task.

10. **Visualize Predictions:**
    - Display model predictions on a small batch of validation images.
    - Compare predicted and actual values for age, gender, and ethnicity.

11. **Generate Grad-CAM Visualizations:**
    - Apply Grad-CAM to the final convolutional layer.
    - Visualize attention regions for:
      - Age prediction
      - Gender classification
      - Ethnicity classification

7. Real-Time Validation Instructions

1. Make sure these files are in the same folder:
   - `live_predict.py`
   - `mobilenet_model.h5`
2. Open Anaconda Navigator or a terminal.
3. Activate your environment:

```
conda activate python10
```

4. Run the script:

   ```
   python live_predict.py
   ```

5. Once the script starts:

   - Your webcam will turn on
   - A window will open showing your face
   - On the screen, you'll see predictions for:
     - **Age**
     - **Gender** (Male or Female)
     - **Ethnicity** (White, Black, Asian, Indian, Others)

6. Press the `q` key on your keyboard to exit.