National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Jeson Varghese Varghese
Student ID: x23254998

School of Computing
National College of Ireland

Supervisor: David Hamill

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Jeson Varghese Varghese
Student ID:	x23254998
Programme:	MSc Data Analytics
Year:	2025
Module:	MSc Research Project
Supervisor:	David Hamill
Submission Due Date:	01/09/2025
Title:	Configuration Manual
Word Count:	655
Page Count:	4

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:	Jeson Varghese
Date:	01/09/2025

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

Jeson Varghese Varghese
x23254998

1 Prelude

This document outlines how to configure and run a SimCLR self-supervised pipeline on CIFAR-10, followed by linear evaluation and a supervised baseline for comparison. It explains the software environment, core parameters, execution issues.

2 System Requirements

2.1 Hardware Requirements

- **GPU:** NVIDIA GPU strongly recommended ($\geq$ 6 GB VRAM). (CPU-only is supported but substantially slower.)
- **CPU:** Modern quad-core or better.
- **RAM:** $\geq$ 8 GB ($\geq$ 16 GB recommended for smoother data loading).
- **Disk Space:** 5 GB for dataset and checkpoints.

2.2 Software Requirements

- **Python:** Version 3.9 or higher.
- **PyTorch:** Version 2.1 or newer (with CUDA build if using GPU).
- **Key Packages:** `torch`, `torchvision`, `numpy`, `pandas`, `matplotlib`, `seaborn`.

3 Using Google Colab/Kaggle

1. Open a new Colab notebook.
2. Install any missing libraries:

```
%pip install torch torchvision torchaudio --index-url https://
   download.pytorch.org/whl/cu121
%pip install numpy pandas matplotlib seaborn
```

3. Upload this notebook or mount a Drive folder for persistent data and results directories.

4 Install Locally

Create and activate a virtual environment:

```
python -m venv .venv
source .venv/bin/activate
```

Install dependencies:

```
pip install torch torchvision torchaudio --index-url https://download.pytorch.
    org/whl/cu121
pip install numpy pandas matplotlib seaborn
```

5 Configuration

Configuration is defined in a `Config` class in the notebook.

Table 1: Configuration

Key	Default	Description
`data_dir`	`./data`	Where CIFAR-10 is stored or downloaded.
`results_dir`	`./results`	Where checkpoints and `summary.csv` are written.
`batch_size`	`512`	Mini-batch size; lower on limited VRAM (e.g., 128).
`simclr_epochs`	`100`	Epochs for SimCLR pretraining.
`sup_epochs`	`100`	Epochs for supervised baseline.
`lr`	`1e-3`	Base learning rate for the optimisers used.
`temperature`	`0.5`	NT-Xent temperature for SimCLR.
`device`	`auto`	`cuda` if available, otherwise `cpu`.
`seed`	`42`	Reproducibility seed.
`num_workers`	`4`	DataLoader workers; set 0 on Windows if hanging.
`label_fractions`	`[0.01, 0.05, 0.1]`	Fractions of labeled data used for linear eval and baseline.
`augmentations`	`[none, weak, strong]`	Training augmentation regimes.

6 Augmentations

`get_transform(aug_type)` defines three pipelines:

- **none** → `ToTensor()` + `Normalize(mean, std)`
- **weak** → Light spatial jitter (e.g., `RandomCrop(32, padding=4)`, `RandomHorizontalFlip()`), then `ToTensor()` + `Normalize(...)`.
- **strong** → `RandomResizedCrop(32, scale=(0.2, 1.0))`, `RandomHorizontalFlip()`, `ColorJitter(...)`, `RandomGrayscale(p=0.2)`, `GaussianBlur(3)`, then `ToTensor()` + `Normalize(...)`.

7 Checkpoints & File Naming

The experiment function writes/loads checkpoints under `results_dir` using these names:

- *SimCLR pretrain:* `simclr_{INT_FRAC}_{AUG}.pt`
  e.g., `simclr_5_weak.pt` for 5% labels + weak augmentation.
- *Supervised baseline:* `sup_{INT_FRAC}_{AUG}.pt`

A summary of results is also saved as:

- **CSV:** `<results_dir>/summary.csv` with columns: `fraction`, `augmentation`, `simclr_accuracy`, `supervised_accuracy`.

8 How to Run

1. Open the notebook and run cells top-to-bottom.
2. Adjust configuration by editing the `Config` values before running the sections titled *Batch Experiment Runner*.
3. The code will iterate over all combinations of `Config.label_fractions` × `Config.augmentations`, performing:

- SimCLR pretraining.
- Linear evaluation.
- Supervised baseline training.
- Writing a row to `summary.csv` for each combination.

9 Data & Directories

Dataset: CIFAR-10 (automatically downloaded by `torchvision` on first run).
Default directories (from Config):

- `data_dir='./data'` → CIFAR data will be stored here.
- `results_dir='./results'` → checkpoints and `summary.csv` are written here.

```
results
|-- simclr_5_weak.pt     # example: 5 percent labels with weak augmentation
|-- sup_5_weak.pt        # supervised baseline checkpoint
‘-- summary.csv          # fraction, augmentation, simclr_accuracy,
   supervised_accuracy
```

These folders are created automatically if missing.

10 Troubleshooting

- **CUDA out of memory:** Lower `Config.batch_size` (e.g., 512 → 256 → 128). Close other GPU applications.
- **Slow on CPU:** Reduce epochs; use a smaller batch size.
- **Very low accuracy:** Ensure normalisation stats were computed and applied; verify that test transforms are deterministic; confirm label-fraction sampling is stratified per class.
- **Need determinism:** Set `torch.backends.cudnn.deterministic = True` and `benchmark = False` (with a potential speed trade-off).

11 CUDA Errors

- Ensure that PyTorch is installed with CUDA support. Install the correct version by visiting the PyTorch Installation Guide.
- Check GPU availability using the following Python command:

```
import torch
print(torch.cuda.is_available())
```

- If `torch.cuda.is_available()` returns `False`, verify that:
  - The system has a compatible NVIDIA GPU.
  - NVIDIA drivers and CUDA toolkit are installed and up-to-date.

12 Resources

12.1 PyTorch Installation Guide

The official PyTorch installation instructions are available here: `https://pytorch.org/get-started/locally/`

12.2 Dataset

The CIFAR-10 dataset used in this project is available at: `https://www.cs.toronto.edu/~kriz/cifar.html`