National College of Ireland

Configuration Manual

MSc Research Project
MSc in Data Analytics (MSCDAD_C)

Karthik Kota
Student ID: X23254653

School of Computing
National College of Ireland

Supervisor: Sallar Khan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Karthik Kota
Student ID:	X23254653
Programme:	MSc in Data Analytics **Year:** 2025
Module:	Research Practicum Part 2
Lecturer:	Sallar Khan
Submission Due Date:	11th Aug 2025
Project Title:	Hybrid Spatiotemporal GraphWaveNet for Real-Time Traffic Forecasting on Dynamic Graphs
Word Count:	1152 **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: Karthik Kota

Date: 11th Aug 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

Karthik Kota
Student ID: x23254653

1 System Requirements

Every experiment is reproducible on a 16 GB MacBook Air using Apple-silicon (M-series) but without the availability of a discrete GPU. PyTorch 2.x allows the on-chip vector units of the CPU to be transparently targeted by using the Apple Accelerate framework, resulting in SIMD-optimised, linear-algebra performance that is sufficient on mid-sized tensors, with low power draw (Apple Inc., 2024; PyTorch Foundation, 2023). Raw SCATS logs, intermediately NumPy arrays and model checkpoints should ideally have about 10 GB of free SSD space, the Unix-standard toolchain in macOS meets all resource requirements at the native-code build level.

2 Required Software and Tools

The project is executed in a clean Python 3.10 environment on macOS (Sonoma 14.x) with a bare minimum library stack: PyTorch 2.2.2, numpy 1.26.4, pandas 2.2.2, dask 2024.8.2, scikit-learn 1.5.1, matplotlib 3.9.2, dash 3.1.0, plotly 6.2.0, jupyterlab 4.2.5, notebook 7 These exact versions (e.g. python -m pip install torch==2.2.2 pandas==2.2.2 ... dash==3.1.0) installed in any new Python3.10 virtual-env or Conda environment will perfectly replicate all of the experiments (Dask Developers, 2024; PyTorch Foundation, 2023).

3 Dataset Preparation

Raw traffic files (SCATSJanuary2024.csv, SCATSFebruary2024.csv) and the fixed signal inventory data are loaded via a clever loader which defaults to pandas but switches to Dask partitions when file size would consume all the available memory. Those timestamps are normalised, gaps less than 6 h are filled forward, and longer outages masked, with any observation over the 99.5th percentile being clipped. All numerical features are then normalized to [0, 1] by a Min-Max scaler, and are followed by sinusoidal hour-of-day and day-of-week embeddings. To start with same and clean set of tensors, four artefacts X.npy, mask.npy, coords.csv, adj.npy are provided so that subsequent notebooks can pick them up (Dask Developers, 2024).

4 Graph Construction

A hybrid adjacency encodes spatial structure. The graph corresponds to a static 9-nearest-neighbour (9-NN) graph based on Haversine distance to a geography prior. Each timestep updates a dynamic layer: node features are linearly projected, followed by one or two KD-trees providing the nearest neighbours, using cosine or RBF similarity with the top-k (k =

16). KD-tree queries are O(NlogN) so the end-to-end latency increases linearly but near constant as the number of nodes doubles (Wikipedia, 2024).

5 Model Setup

The forecasting engine is based on Graph WaveNet, a method of coupling dilated temporal convolutions and diffusion-based spatial convolutions on an adaptive adjacency matrix (Wu et al., 2019). It provides three varieties namely (i) a stationary baseline, (ii) SE-FB hybrid, which updates edge weights, either with cosine or RBF similarity to update edge weights (iii) reference baselines including AGCRN. All models are trained on masked MSE, Adam (LR = 0.001), batch size 64, early stopping after 15 non-improving epochs. It provides three operating modes: H3 (3-step one-shot), H24-AR (24-step autoregressive) and H24-TOD (24-step one-shot with time-of-day cues).

6 Dashboard Notebook

Running 04_reporting_dashboard.ipynb starts a local Dash server and reads artefacts/metrics.json then displays KPI cards, horizon-wise error bars, latency-versus-accuracy scatter plots and a dropout-robustness slider. The fact that the dashboard only utilizes previously calculated logs means that it can be responsive even on a laptop processor and does not need retraining as it is explored.

7 Project Structure

The repository is laid out so that a newcomer can reproduce the entire workflow by following four root notebooks—01_data_pipeline.ipynb (ETL and feature engineering), 02_model_training.ipynb (building and fitting all model variants), 03_model_evaluation.ipynb (benchmarking and logging), and 04_reporting_dashboard.ipynb (launching an interactive Plotly-Dash UI)—while all reusable tensors (X.npy, mask.npy, coords.csv, adj.npy) and a deterministic split map (split_idx.json) sit alongside them for immediate reloads; model-specific folders (static_gwn/, agcrn/, se_fb_cosine/, se_fb_rbf/, etc.) each contain their own source file plus a best checkpoint (_best.pth), and evaluation artefacts funnel into artefacts/metrics.json (with a summary table in model_leaderboard.csv) that the dashboard consumes, whereas publication graphics reside in figures/, raw inputs remain in the three SCATS CSVs (plus two merged Parquet snapshots for rapid EDA), and a top-level README.md documents the exact shell commands required to replay every stage making the codebase fully self-contained and reproducible.

8 Logs and Evaluation

All training runs record RMSE, MAE, MAPE, wall-clock inference latency and change in error with 05 % sensor dropout in the data. Timing incorporates time.perf_counter() so that build-time of the KD-tree and forward-time of the neural can be timed separately, as is essential when diagnosing performance on CPU-only systems. The various outputs all converge to the same metrics.json so that there is a truth shared downstream between the dashboard and any subsequent scripts.

9 Troubleshooting

Memory Error during CSV load → allow the loader to switch automatically to Dask or pre-split the file.

Missing tensors (adj.npy) → rerun the data-pipeline notebook to regenerate aligned artefacts.

Blank dashboard → refresh the browser after the kernel completes; Dash hot-reloads once files settle.

Latency spikes at high node counts → lower k in the KD-tree or schedule batch inference during off-peak hours.

References

Apple Inc. (2024) *Accelerate Overview*. Available at: https://developer.apple.com/accelerate/.

Dask Developers (2024) *Dask DataFrame Documentation*. Available at: https://docs.dask.org/en/stable/dataframe.html.

PyTorch Foundation (2023) 'PyTorch 2.1 Release Notes'. Available at: https://pytorch.org/blog/pytorch-2-1/.

Wikipedia (2024) 'k-d Tree'. Available at: https://en.wikipedia.org/wiki/K-d_tree.

Wu, Z. *et al.* (2019) 'Graph WaveNet for Deep Spatial-Temporal Graph Modelling', *Proceedings of IJCAI-19*. Available at: https://arxiv.org/abs/1906.00121 .