
Master Thesis

# Thesis Title: Concise and Engaging Title

by

**Mateusz Kędzia**
(2666752)

*Supervisor*: Ronald Siebes (VU Amsterdam)
*Daily Supervisor*: Jiancheng Weng (Beijing University of Technology)
*Internal Advisor*: Zhisheng Huang (VU Amsterdam)
*External Advisor*: Shuai Wang (VU Amsterdam/Maastricht University)
*Second Reader*: Name and Surname

June 4, 2025

Submitted in partial fulfillment of the requirements for
the VU degree of Master of Science in Artificial Intelligence

# Knowledge Distillation for Map-Matched Trajectory Prediction: Improving Urban Route Prediction through Cross-Task Knowledge Transfer

Mateusz Kędzia[1][0009−0001−4296−4479]

Vrije Universiteit Amsterdam, Amsterdam

**Abstract.** Urban traffic management, transportation planning, and intelligent city systems require accurate real-time trajectory prediction to support policy decisions and optimize traffic flow. However, existing fast prediction models suffer from poor route completion rates (12-18%), limiting their practical deployment for traffic regulators and urban planners. While sophisticated models like LM-TAD achieve superior spatial reasoning, their computational overhead ( 3.4ms per trajectory vs 0.1ms for fast models) prevents real-time application in city-wide traffic management systems, digital twin platforms, and large-scale simulations.

This thesis addresses this challenge through training-time knowledge distillation, transferring spatial understanding from LM-TAD (a trajectory anomaly detection model) to HOSER (a fast zone-based prediction model). We demonstrate that repurposing the "normal trajectory" knowledge learned by anomaly detection models enables dramatic improvements in route prediction without inference-time overhead. Our distillation framework achieves 85-89% path completion success (47-74× improvement over vanilla baseline), 87% better distance distribution matching, and 98% better spatial pattern fidelity on Beijing's 40,060-road network with 629,380 training trajectories. Hyperparameter optimization reveals that minimal distillation weight ($\lambda$=0.0014) with high temperature ($\tau$=4.37) enables effective knowledge transfer while preserving the student model's fast inference speed.

The resulting system enables practical deployment for policy makers and traffic regulators, supporting applications in real-time traffic signal optimization, infrastructure planning, urban digital twins, agent-based traffic simulation, and high-quality synthetic trajectory data generation for training other models. This work demonstrates the viability of cross-task knowledge distillation for trajectory prediction and provides a scalable framework for integrating AI-based route prediction into operational traffic management systems.

**Keywords:** Knowledge distillation · Trajectory prediction · Urban transportation · Traffic management · Digital twins · Deep learning

## 1 Introduction

Accurate trajectory prediction is foundational to intelligent transportation systems, underpinning applications from dynamic navigation and fleet dispatch to digital-twin simulation of urban flow. Modern cities contain tens of thousands of interconnected road segments; a practical predictor must therefore reason over large graphs while delivering sub-second latency at metropolitan scale.

State-of-the-art transformer models excel at learning long-range spatial dependencies, yet their quadratic self-attention incurs inference times incompatible with real-time traffic management. Conversely, lightweight graph-aware models such as HOSER achieve millisecond-level speed but fall short in route-completion accuracy. This accuracy–latency dichotomy poses a central research challenge: *how can one inherit the rich spatial knowledge of heavy models without deploying them at run time?*

This thesis answers the question by distilling the transformer-based LM-TAD anomaly detector into the hierarchical, low-latency HOSER predictor *during training only*. Our cross-task distillation transfers spatial priors learned in anomaly detection to next-step prediction, yielding a student that approaches transformer accuracy while preserving operational efficiency.

*Contributions.* We make four key contributions:

- Propose the first cross-task distillation framework that transfers spatial knowledge from trajectory anomaly detection to trajectory prediction.
- Develop a batched, GPU-optimised KL divergence module that enables large-scale training with negligible overhead.
- Empirically validate the distilled model on large-scale urban datasets, showing pronounced improvements across established mobility-generation metrics while maintaining millisecond-level inference latency.
- Release an end-to-end evaluation pipeline for trajectory generation and similarity analysis, facilitating reproducible research.

*Paper organisation.* section 2 surveys the evolution of trajectory modelling, culminating in the need for knowledge distillation. section 3 details the LM-TAD $\rightarrow$ HOSER distillation algorithm and training pipeline. section 5 describes dataset preparation, and section 6 presents empirical results. We conclude with future research directions in section 7.

## 2 Related Work

This section reviews the key research areas relevant to knowledge distillation for trajectory prediction. We examine trajectory prediction and generation methods, synthetic data applications for urban systems, trajectory anomaly detection approaches that learn spatial patterns, knowledge distillation techniques, the architectural foundations in graph neural networks and transformers, and cross-task knowledge transfer methods.

The evolution of data-driven trajectory research mirrors the broader progression of sequential modelling in artificial intelligence. Beginning with recurrent neural networks and hand-crafted statistical baselines, the field has steadily adopted richer spatial representations, more expressive sequence architectures and, most recently, cross–task knowledge transfer. The subsections below follow this chronological arc, critiquing the advantages and limitations of each paradigm to motivate the distillation framework presented in section 3.

### 2.1 Classical Trajectory Prediction Models

Early work framed next–location prediction as a sequence modelling problem amenable to recurrent neural networks (RNNs) and their gated variants. Memory-augmented LSTMs [14] and variational autoencoders [15] captured short-range dependencies and route uncertainty, while GAN-based approaches such as LSTM-TrajGAN [18] attempted adversarially faithful path synthesis. These models established the feasibility of learning spatial–temporal patterns directly from GPS data, but their limited receptive field and difficulty handling map constraints curbed real-world adoption. *Takeaway*: classical RNN/GAN models prove the concept yet struggle with long-range coherence and graph topology.

### 2.2 Graph Neural Networks for Road Networks

The introduction of graph neural networks (GNNs) [10,20] provided an explicit inductive bias for road topology. By representing intersections and road segments as nodes and edges, GCNs aggregated neighbourhood information, whereas GATs learned edge-specific attention weights, supporting fine-grained routing decisions. Hierarchical hybrids further combined local and regional reasoning. Despite clear spatial benefits, pure GNN solutions often incurred high inference latency on large urban graphs. *Takeaway*: GNNs embed topology elegantly but computational cost motivates search for lighter yet expressive alternatives.

### 2.3 Transformer Architectures for Mobility Sequences

Transformers [19] revolutionised sequential learning through self-attention. Mobility research quickly adopted this paradigm—LM-TAD treats trajectories as token sequences, achieving state-of-the-art anomaly detection accuracy. Large language model (LLM) adaptations such as PathGen-LLM [13] demonstrated zero-shot path generalisation. However, the quadratic cost of self-attention renders vanilla transformers impractical for real-time traffic services. *Takeaway*: transformers learn rich global dependencies but impose prohibitive inference overhead.

### 2.4 Deep Generative Approaches: VAEs to Diffusion and LLMs

Parallel to architectural advances, generative modelling transitioned from VAEs to diffusion processes and LLM-style decoders. Diffusion-based TrajGDM [5] reframed generation as uncertainty reduction, yielding diverse and realistic paths. TrajGPT [9] leveraged transformers for controllable synthesis, while foundation-scale models integrated multi-modal context [16]. These methods improved fidelity but further increased computational and data demands. *Takeaway*: modern generative models capture complex mobility patterns yet exacerbate scalability concerns.

*Road so far.* Classical RNNs established learnability, GNNs injected topology awareness, transformers unlocked long-range context, and diffusion/LLM generators raised realism. Nevertheless, none reconcile spatial expressiveness with the latency constraints of traffic operations.

### 2.5 Synthetic Trajectory Generation for Urban Applications

Recent literature positions synthetic data as a utility-centric asset for simulation and policy rather than solely for privacy. SynMob [21] and related frameworks retain geo-statistical properties critical to urban planning. This shift underlines the importance of high-quality generation that scales across cities, a requirement echoed by foundation mobility models [16]. *Takeaway*: urban stakeholders demand scalable, high-fidelity synthesis, heightening the need for efficient yet accurate predictors.

### 2.6 Trajectory Anomaly Detection and Spatial Learning

Anomaly detectors such as LM-TAD learn what *normal* mobility looks like by modelling probability distributions over location tokens [7,11]. Their spatial insight is invaluable, yet transformer-based detectors are slower than prediction-oriented models like HOSER. *Takeaway*: anomaly detection encodes rich spatial priors that remain untapped by fast predictors.

### 2.7 Knowledge Distillation and Model Compression

Knowledge distillation [8] addresses the accuracy–efficiency dilemma by transferring soft targets from a high-capacity teacher to a lightweight student. Vision and NLP studies show that students can approximate teachers with negligible runtime overhead. *Takeaway*: distillation offers a principled route to inherit transformer knowledge without paying inference cost.

### 2.8 Cross-Task Transfer and Foundation Mobility Models

Cross-task transfer extends distillation across objectives—e.g. leveraging anomaly-detection priors to boost prediction [16]. Foundation mobility models exemplify multi-domain knowledge sharing, yet concrete methods for teacher–student bridging remain under-explored. *Takeaway*: transferring spatial knowledge across tasks is promising but lacks systematic methodologies.

*Road so far.* The literature converges on two complementary insights: (i) transformers learn superior spatial representations, and (ii) operational systems require millisecond-scale inference.

#### Synthesis and Motivation for This Thesis

Bridging these insights, our work distils the transformer-based LM-TAD anomaly detector into the graph-aware, low-latency HOSER predictor during training only, as detailed in section 3. This approach inherits rich spatial priors while preserving real-time performance, directly addressing the shortcomings identified above.

## 3 Methodology

This section presents our knowledge distillation framework for transferring spatial reasoning from a trajectory anomaly detector to a fast route predictor. Figure 1 illustrates the complete pipeline, showing how the frozen teacher and trainable student interact through vocabulary mapping and temperature-scaled distributions. We detail the vocabulary alignment mechanism (subsection 3.2), the temperature-scaled distillation objective (subsection 3.3), the optimized training pipeline (subsection 3.4), and the inference procedure (subsection 3.5).

### 3.1 Preliminaries

**Notation** Table 1 summarizes the mathematical notation used throughout this section.

#### Problem Definition

**Definition 1 (Trajectory).** *A* trajectory $T$ *is a sequence of road segments:*

$$T = \{r_1, r_2, \ldots, r_n\} \quad \textit{where } r_i \in \mathcal{V}$$

*representing a path through the road network* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

**Definition 2 (Timestamped Trajectory).** *A* timestamped trajectory $T$ *includes arrival times at each road segment:*

$$T = \{(r_1, t_1), (r_2, t_2), \ldots, (r_n, t_n)\} \quad \textit{where } r_i \in \mathcal{V}, t_i \in \mathbb{R}_+$$

*with timestamps satisfying* $t_1 < t_2 < \cdots < t_n$.

**Table 1.** Mathematical notation

| Symbol | Description |
|---|---|
| $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ | Road network graph with $\lvert\mathcal{V}\rvert$ segments |
| $\mathbf{r}_{1:t} = (r_1, \ldots, r_t)$ | Partial trajectory of $t$ road segments |
| $\mathcal{C}_t \subseteq \mathcal{V}$ | Candidate set at timestep $t$ (top-$k$ roads) |
| $\mathcal{Z} = \{1, \ldots, V\}$ | Grid cell vocabulary |
| $\mathcal{L}_\phi : \mathcal{Z}^w \to \Delta^{\lvert\mathcal{Z}\rvert}$ | Teacher model (frozen) |
| $\mathcal{H}_\theta : \mathcal{V}^t \times \mathcal{V} \to \Delta^{\lvert\mathcal{C}\rvert}$ | Student model (trainable) |
| $\psi : \mathcal{V} \to \mathcal{Z}$ | Cross-vocabulary mapping |
| $q^{(\tau)}, p^{(\tau)}$ | Temperature-scaled distributions |
| $\tau$ | Temperature parameter |
| $\lambda$ | Distillation loss weight |

**Definition 3 (Cross-Vocabulary Mapping).** *The mapping $\psi : \mathcal{V} \to \mathcal{Z}$ assigns each road segment to its containing grid cell:*

$$\psi(v) = \left\lfloor \frac{x_v - x_{\min}}{\Delta_x} \right\rfloor \cdot n_{cols} + \left\lfloor \frac{y_v - y_{\min}}{\Delta_y} \right\rfloor \tag{1}$$

*where $(x_v, y_v) = centroid(v)$ and $\Delta_x, \Delta_y$ are grid resolutions.*

**Definition 4 (Cross-Task Knowledge Distillation).** *Given frozen teacher $\mathcal{L}_\phi$ trained for anomaly detection and trajectory dataset $\mathcal{D} = \{(\mathbf{r}_i, y_i)\}_{i=1}^N$, learn student parameters:*

$$\theta^* = \arg\min_{\theta \in \Theta} \mathbb{E}_{(\mathbf{r},y)\sim\mathcal{D}} \left[ \mathcal{L}_{CE}(\mathcal{H}_\theta(\mathbf{r}), y) + \lambda \mathcal{L}_{KL}^{(\tau)}(\mathcal{L}_\phi \circ \psi(\mathbf{r}), \mathcal{H}_\theta(\mathbf{r})) \right] \tag{2}$$

**Model Specifications** We transfer knowledge from the pre-trained LM-TAD teacher $\mathcal{L}_\phi$ [17] to the HOSER student $\mathcal{H}_\theta$ [4]. The teacher operates on grid cell sequences with vocabulary size $\lvert\mathcal{Z}\rvert = V$, while the student predicts over $\lvert\mathcal{V}\rvert$ road segments.

### 3.2 Vocabulary Alignment

The cross-vocabulary mapping $\psi$ (Definition 3) enables knowledge transfer between the teacher's grid-based representation and the student's road-based representation. The mapping assigns each road segment to its containing grid cell based on centroid coordinates (detailed algorithm in Appendix A.1).

*Remark 1 (Many-to-One Mapping).* Multiple roads may correspond to a single grid cell, with higher density in urban cores.

**Probability Renormalization** Since the teacher produces distributions over all grid cells $\mathcal{Z}$ while the student operates over candidate roads $\mathcal{C}_t$, we extract and renormalize:

$$\tilde{q}_t(c) = q(\psi(c) \mid \mathbf{z}_{1:t}) \quad \forall c \in \mathcal{C}_t \tag{3}$$

$$q_t^{(\tau)}(c) = \frac{\exp(\log \tilde{q}_t(c)/\tau)}{\sum_{c' \in \mathcal{C}_t} \exp(\log \tilde{q}_t(c')/\tau)} \tag{4}$$

### 3.3 Knowledge Distillation Framework

**Temperature Scaling**

**Theorem 1 (Temperature-Scaled Knowledge Transfer).** *Given teacher logits $\boldsymbol{\ell}^{\mathcal{L}} \in \mathbb{R}^{|\mathcal{C}|}$ and student logits $\boldsymbol{\ell}^{\mathcal{H}} \in \mathbb{R}^{|\mathcal{C}|}$, the distillation loss is:*

$$\mathcal{L}_{KL}^{(\tau)} = \tau^2 \sum_{i \in \mathcal{C}_t} q_i^{(\tau)} \log \frac{q_i^{(\tau)}}{p_i^{(\tau)}} \tag{5}$$

*where:*

$$q_i^{(\tau)} = \frac{\exp(\ell_i^{\mathcal{L}}/\tau)}{\sum_{j \in \mathcal{C}_t} \exp(\ell_j^{\mathcal{L}}/\tau)} \tag{6}$$

$$p_i^{(\tau)} = \frac{\exp(\ell_i^{\mathcal{H}}/\tau)}{\sum_{j \in \mathcal{C}_t} \exp(\ell_j^{\mathcal{H}}/\tau)} \tag{7}$$

*The $\tau^2$ scaling factor preserves gradient magnitudes as $\tau$ increases.*

*Proof.* The gradient with respect to student logit $\ell_i^{\mathcal{H}}$ is:

$$\frac{\partial \mathcal{L}_{\text{KL}}}{\partial \ell_i^{\mathcal{H}}} = \frac{1}{\tau}(p_i^{(\tau)} - q_i^{(\tau)}) \tag{8}$$

which scales as $\mathcal{O}(1/\tau)$. Multiplying by $\tau^2$ ensures $\mathcal{O}(\tau)$ scaling, preventing gradient vanishing in the high-temperature regime where distillation is most effective [8].

**Combined Training Objective** The total loss combines supervised learning with knowledge distillation:

$$\mathcal{L}_{\text{total}} = \underbrace{\mathcal{L}_{\text{CE}}(p, y)}_{\text{hard targets}} + \alpha \underbrace{\mathcal{L}_{\text{time}}(\hat{t}, t)}_{\text{auxiliary}} + \lambda \underbrace{\mathcal{L}_{\text{KL}}^{(\tau)}(q^{(\tau)}, p^{(\tau)})}_{\text{soft targets}} \tag{9}$$

where $\alpha$ is fixed and $\lambda$ is tuned to balance supervised and distillation objectives.

### 3.4 Training Pipeline

The teacher parameters $\phi$ remain frozen throughout training. Per-iteration complexity is $\mathcal{O}(B \cdot t \cdot (V + k^2))$ where teacher inference dominates at $\mathcal{O}(B \cdot t \cdot V)$. Implementation details are provided in section 4.

**Algorithm 1** Knowledge Distillation Training

**Require:** Dataset $\mathcal{D}$, Teacher $\mathcal{L}_\phi$, Student $\mathcal{H}_\theta$, hyperparameters $\{\lambda, \tau, w, \eta\}$
**Ensure:** Trained parameters $\theta^*$

Initialize $\theta$ from normal distribution
**for** epoch $= 1$ to $E$ **do**
  **for** $(\mathbf{r}, y) \in \mathcal{D}$ **do**
    $\mathbf{z} \leftarrow \psi(\mathbf{r})$ ▷ Map roads to grid cells
    $\mathbf{q} \leftarrow \mathcal{L}_\phi(\mathbf{z}_{t-w:t})$ ▷ Teacher inference (no gradient)
    $q^{(\tau)} \leftarrow \text{Renormalize}(\mathbf{q}, \mathcal{C}_t, \tau)$ ▷ Eq. (4)
    $\boldsymbol{\ell} \leftarrow \mathcal{H}_\theta(\mathbf{r}, \mathcal{C}_t)$ ▷ Student logits
    $p^{(\tau)} \leftarrow \text{Softmax}(\boldsymbol{\ell}/\tau)$
    $\mathcal{L} \leftarrow \mathcal{L}_{\text{CE}}(\boldsymbol{\ell}, y) + \lambda \mathcal{L}_{\text{KL}}^{(\tau)}(q^{(\tau)}, p^{(\tau)})$
    $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}$
  **end for**
**end for**
**return** $\theta^*$

### 3.5 Inference and Evaluation

At inference time, we employ only the trained student $\mathcal{H}_{\theta^*}$ with beam search to generate trajectories. Given origin-destination pair $(r_o, r_d)$, the student produces complete trajectories efficiently.

We evaluate using global distribution metrics (Jensen-Shannon Divergence) and local trajectory metrics (Hausdorff, DTW, EDR) computed separately on training and test OD pairs. Full evaluation details are provided in subsection 6.2.

## 4 Implementation

This section describes the practical implementation of our knowledge distillation framework, covering dataset preparation, hyperparameter tuning, training optimizations, and computational infrastructure. These implementation choices enable efficient training on commodity hardware while maintaining reproducibility.

### 4.1 Codebase and Modifications

Our experiments build upon the official implementations of both LM-TAD [17,2] and HOSER [4,1]. We modified both codebases to integrate the distillation framework while preserving the original model architectures and hyperparameters from their respective papers. Both vanilla HOSER (baseline) and distilled HOSER (proposed) variants run on our unified modified implementation to ensure fair comparison—the distillation loss ($\lambda \mathcal{L}_{\text{KL}}$) is the only difference between the two configurations. This controlled setup isolates the impact of knowledge transfer from other implementation factors.

Additional engineering optimizations include: replacing HOSER's A* greedy search with beam search for trajectory generation (improved efficiency), vectorizing data collation functions, and adding WandB integration for experiment tracking. These modifications improve computational performance without affecting model behavior or evaluation fairness.

### 4.2 Dataset Preparation Pipeline

The distillation framework requires preprocessing to bridge HOSER's road-based and LM-TAD's grid-based representations. We use the Beijing dataset (subsection 5.1) from the original HOSER paper [4].

**Road Network Preprocessing** The student model operates on hierarchical zones as specified in the original HOSER paper [4]. Zone partitioning and transition matrix construction follow the procedures detailed in subsection 5.4.

**Teacher Model Preparation** The LM-TAD teacher model is loaded from pre-trained checkpoints with vocabulary mapping $\psi$ (Definition 3) precomputed and cached for efficient training. Checkpoint conversion details are provided in subsection 5.4.

### 4.3 Hyperparameter Optimization

We employ a systematic two-phase hyperparameter search using the Optuna framework [3] to identify optimal distillation parameters.

**Search Space Design** Three hyperparameters govern knowledge transfer (Table 6, Appendix A.3): $\lambda$ (distillation weight, log scale $[0.001, 0.1]$) controls teacher influence vs. supervised signal; $\tau$ (temperature, linear $[1.0, 5.0]$) smooths distributions to expose "dark knowledge"; and $w$ (window size, $[2, 8]$) determines teacher context length, trading accuracy for speed.

**Two-Phase Optimization Strategy Phase 1: Exploration (12 trials).** We employ CMA-ES (Covariance Matrix Adaptation Evolution Strategy) [6] as the sampler, which efficiently navigates continuous parameter spaces. The Hyperband pruner [12] with minimum resource allocation of 5 epochs terminates unpromising configurations early.

Key configuration:

- **Objective**: Maximize validation accuracy after 8 epochs
- **Baseline**: Trial 0 always runs vanilla training ($\lambda = 0$) for fair comparison
- **Budget**: 12 trials $\times$ 8 epochs = 96 training runs (pruning reduces actual compute)

**Phase 2: Validation (3 seeds).** The best configuration from Phase 1 is trained to completion (25 epochs) with three random seeds $\{42, 43, 44\}$ to assess robustness and estimate variance.

**Optimal Configuration** The hyperparameter search yields an unexpected result: minimal distillation weight with high temperature proves most effective.

**Table 2.** Optimal distillation hyperparameters from Optuna tuning

| Parameter | Optimal Value | Interpretation |
|---|---|---|
| $\lambda$ | 0.0014 | Very subtle teacher guidance |
| $\tau$ | 4.37 | High temperature (broad knowledge transfer) |
| $w$ | 7 | Nearly full context window |

This configuration suggests that *subtle distributional guidance* from the teacher, rather than aggressive knowledge transfer, enables the student to integrate spatial understanding without compromising its architectural strengths.

### 4.4 Training Optimizations

To enable practical training on commodity GPU hardware, we implement several key optimizations that reduce memory consumption and accelerate training.

**Automatic Mixed Precision** We employ PyTorch's Automatic Mixed Precision (AMP) framework to reduce memory footprint:

- **Teacher inference**: FP16 precision with `autocast` context reduces memory by ∼50% with negligible accuracy loss
- **Student training**: TF32 format for matrix operations maintains numerical stability
- **Gradient scaling**: Dynamic loss scaling prevents underflow in FP16 backward passes

The frozen teacher benefits most from FP16 inference, as it performs no gradient updates and requires only forward-pass accuracy.

**Memory Management Intelligent Caching.** The framework automatically decides whether to cache the dataset in RAM based on available memory. For the Beijing dataset (∼630k trajectories, ∼13GB), RAM caching eliminates disk I/O bottlenecks when sufficient memory is available. The dataset can be loaded entirely into RAM alongside model and optimizer states. Otherwise, the system streams data from NVMe storage with minimal performance degradation.

**Gradient Accumulation.** We simulate an effective batch size of 512 by accumulating gradients over 8 micro-batches of 64 samples each. This enables large-batch training benefits while respecting GPU memory constraints.

**Candidate Filtering.** HOSER's spatial pruning limits the candidate set to $k = 64$ nearest roads per timestep, reducing the output dimensionality from $|\mathcal{V}| = 40{,}060$ to a manageable subset.

**Batched Operations** All vocabulary mapping and teacher inference operations are fully vectorized:

- **Road-to-grid mapping**: GPU-accelerated tensor operations via precomputed lookup tables
- **Label remapping**: Parallel remapping with masked positions set to $-100$ (ignored by loss)
- **Teacher inference**: Batch-wise forward pass processes all timesteps simultaneously

These optimizations yield 11–13 iterations/second training throughput. Crucially, teacher inference adds less than 2% overhead compared to vanilla HOSER training, making distillation nearly cost-free.

### 4.5 Training Infrastructure

**Hardware:** NVIDIA RTX 2080 Ti GPU with 64GB system RAM (Appendix A.3 for complete specifications).

**Training Configuration** We use AdamW optimizer with cosine annealing and effective batch size 512 (64 samples $\times$ 8 accumulation steps). Training runs 25 epochs ($\sim$36 hours) with fixed seeds (42, 43, 44) for reproducibility. Full configuration details are provided in Table 7, Appendix A.3.

### 4.6 Practical Considerations

**Dataset-Specific Adaptations** Different urban networks require minor adaptations:

**Beijing**: Standard configuration with 1024 max trajectory length.

**Porto**: Longer trajectories (avg. 8 vs 4.6 points) require reduced batch size ($64 \rightarrow 32$) and gradient checkpointing to prevent memory overflow. Memory usage scales quadratically with trajectory length due to attention mechanisms and distance matrices.

**BJUT (Private Beijing)**: Similar to Beijing reference dataset but requires independent map-matching and preprocessing pipeline.

**Common Challenges and Solutions Grid dimension mismatch**: Ensure LM-TAD's `grip_size` parameter matches the vocabulary mapping configuration. Beijing uses $205 \times 252$ grid, Porto uses $46 \times 134$.

**Highway type parsing**: Some datasets use nested list formats (e.g., `["primary", "secondary"]`), others use integer codes. The data loader handles both via conditional parsing.

**Memory overflow**: Reduce batch size or enable gradient checkpointing for datasets with longer trajectories. Porto requires 50% smaller batches than Beijing despite similar dataset sizes.

**Reproducibility:** Complete implementation, trained models, evaluation scripts, and configuration files are provided in the supplementary materials to facilitate reproduction of all experiments.

## 5 Datasets and Preprocessing

This section describes the datasets used for distillation experiments, their preprocessing pipeline, and the compatibility layer required to bridge HOSER's road-based and LM-TAD's grid-based representations. We evaluate our framework on three urban trajectory datasets representing different cities and data sources.

### 5.1 Dataset Overview

We evaluate the distillation framework on three urban trajectory datasets (statistics in Table 8, Appendix A.4). The Beijing HOSER reference dataset [4] provides our primary evaluation benchmark, while Porto and BJUT datasets enable cross-dataset validation.

**Beijing HOSER Reference Dataset:** Standardized benchmark from the original HOSER paper with 40,060 road segments and 629,380 training trajectories. Map-matched by the original authors, ensuring high alignment quality. We adapted the format for LM-TAD compatibility while preserving train/validation/test splits.

**Porto HOSER Dataset:** Different urban environment with longer average trajectories (8.0 vs 4.6 road segments). Also from the original HOSER paper [4]. [EVALUATION IN PROGRESS]

**Beijing Private (BJUT) Dataset:** Independent data source from Beijing University of Technology for cross-dataset validation. [TO BE COMPLETED]

### 5.2 Dataset Format

We use the HOSER dataset format [4], which represents trajectories as sequences of map-matched road segment IDs with corresponding timestamps and explicit origin-destination pairs. The road network topology and attributes enable candidate generation for spatial pruning during prediction. Map-matching was performed by the original HOSER authors, ensuring high alignment quality.

### 5.3 Vocabulary Alignment

LM-TAD tokenizes trajectories using a uniform spatial grid [17], while HOSER predicts over road segments. We construct the mapping $\psi : \mathcal{V} \to \mathcal{Z}$ (Definition 3) that assigns each road to its corresponding grid cell based on centroid coordinates, enabling cross-task knowledge transfer. Grid configuration: Beijing uses $205 \times 252 = 51{,}660$ cells; Porto uses $46 \times 134 = 6{,}164$ cells.

**Vocabulary Alignment Statistics** The vocabulary alignment characteristics (Table 9, Appendix A.4) show Beijing averages 0.78 roads per grid cell. This many-to-one mapping (multiple roads per cell, particularly in dense urban cores) enables the teacher's distributional knowledge over grid cells to inform the student's predictions over road segments, despite the vocabulary size mismatch.

### 5.4 Preprocessing Pipeline

Dataset preparation involves: (1) zone partitioning following HOSER's hierarchical structure [4], (2) zone transition matrix computation from training trajectories, (3) LM-TAD teacher checkpoint preparation [17], and (4) vocabulary mapping construction. These steps adapt the base models for distillation training. Preprocessing is efficient: zone partitioning takes ∼15–20 seconds for Beijing (40,060 roads), and transition matrix construction takes ∼10–15 seconds (629,380 trajectories).

**Data Splitting and OD Stratification** HOSER's original data splits are preserved to maintain comparability:

- **Training set**: Used for distillation and model parameter updates
- **Validation set**: Used for hyperparameter tuning (Optuna objective)
- **Test set**: Held out for final evaluation (never seen during training or tuning)

Critically, these splits are stratified by origin-destination (OD) pairs. The test set contains OD pairs *not present in training*, enabling evaluation of generalization to unseen routes rather than mere memorization.

### 5.5 Dataset Statistics

Complete dataset statistics are provided in Table 8, Appendix A.4. Key characteristics:

- **Beijing**: 40,060 roads, 629,380 training trajectories, average length 4.6 roads, 5.16 km trips
- **Porto**: ∼11,024 roads, ∼481,359 training trajectories, average length 8.0 roads (substantially longer)
- **BJUT**: Planned for cross-dataset validation (independent preprocessing)

**Note on trajectory length:** Porto trajectories are substantially longer than Beijing (8.0 vs 4.6 road segments), leading to quadratic memory scaling in attention mechanisms. This necessitates reduced batch sizes for Porto experiments (see subsection 4.6).

### 5.6 Data Quality and Representativeness

The datasets represent real-world urban mobility patterns with inherent characteristics:

**Beijing:** High-density metropolitan area with complex road hierarchy. Trajectories span residential, commercial, and transportation hub regions, providing diverse route types.

**Porto:** Smaller city with different urban structure. Longer trajectories suggest different mobility patterns (potentially more suburban routes or lower road density).

**BJUT (planned):** Independent data source enables assessment of whether distillation benefits generalize beyond HOSER's curated benchmark datasets.

All datasets are *map-matched*, meaning GPS points have been aligned to road segments via sophisticated algorithms. This preprocessing step, performed by domain experts, eliminates GPS noise and ensures trajectory realism. Our distillation framework operates entirely on these cleaned, road-level trajectories.

## 6 Experimental Evaluation

This section presents a comprehensive empirical evaluation of knowledge distillation for trajectory prediction. We begin by describing the experimental setup and evaluation metrics (subsection 6.1, subsection 6.2), then present detailed results for the Beijing dataset (subsection 6.3), followed by placeholders for ongoing Porto and planned BJUT evaluations (subsection 6.4, subsection 6.5). We conclude with cross-dataset analysis and inference speed discussion (subsection 6.6, subsection 6.7).

### 6.1 Experimental Setup

**Models Evaluated** We compare two model configurations trained under identical conditions, differing only in whether distillation is enabled:

**Vanilla HOSER (Baseline):** Standard maximum likelihood training without teacher guidance. This corresponds to Trial 0 in our hyperparameter optimization, with distillation weight $\lambda = 0$. The vanilla model learns exclusively from hard labels (ground-truth next roads).

**Distilled HOSER (Proposed):** HOSER student trained with knowledge distillation from the frozen LM-TAD teacher. The optimal configuration from hyperparameter tuning uses $\lambda = 0.0014$, $\tau = 4.37$, and window size $w = 7$. The distilled model learns from both hard labels and soft teacher distributions.

**Fair Comparison Protocol** To isolate the effect of knowledge distillation, all other training parameters remain identical between vanilla and distilled models (Table 7, Appendix A.3). Both use HOSER architecture with AdamW optimizer ($\eta = 5 \times 10^{-4}$), effective batch size 1024 ($128 \times 8$ accumulation steps), 25 epochs with cosine annealing, and identical train/val/test splits. The *only difference* is distillation: vanilla sets $\lambda = 0$ (disabled), while distilled uses $\lambda = 0.0014$, $\tau = 4.37$, $w = 7$.

This controlled experimental design ensures that performance differences stem purely from knowledge transfer, not confounding factors like different batch sizes, learning rates, or architectures.

**Hyperparameter Tuning** Optimal distillation hyperparameters ($\lambda = 0.0014$, $\tau = 4.37$, $w = 7$) were identified via systematic Optuna search (subsection 4.3). Trial 0 establishes the vanilla baseline ($\lambda = 0$), validated with seeds 42, 43, 44.

**Trajectory Generation Protocol** For evaluation, we generate synthetic trajectories using beam search with width $b = 4$:

1. Sample 5,000 origin-destination (OD) pairs from real training set (memorization test)
2. Sample 5,000 OD pairs from real test set (generalization test)
3. For each OD pair, generate a complete trajectory using the trained model
4. Compare generated trajectories against real trajectories with matching OD pairs

This protocol separately assesses *memorization* (train OD) and *generalization* (test OD), revealing whether models merely overfit training patterns or learn transferable spatial reasoning.

### 6.2 Evaluation Metrics

We employ a comprehensive metric suite covering global distribution quality, local trajectory similarity, and path completion capability. Most metrics follow the HOSER evaluation framework [4,1] to enable direct comparison. Our key methodological contribution is the systematic generation and evaluation on both training OD pairs and held-out test OD pairs separately—extending HOSER's approach of sampling OD pairs from the training distribution only. This train/test comparative analysis enables assessment of memorization versus generalization, revealing that distilled models learn transferable spatial reasoning rather than overfitting training routes. We also introduce OD pair matching rate as an explicit metric for path completion success. Detailed formulations are provided in Appendix A.2.

**Global Distribution Metrics** These metrics assess whether aggregate statistics of generated trajectories match real data distributions:

- **Jensen-Shannon Divergence (JSD)**: Symmetric divergence measure (bounded $[0, 1]$) computed for distance, duration, and radius of gyration distributions. Lower values indicate better distribution matching.
- **Radius of Gyration**: Measures spatial spread of trajectories, capturing how geographically dispersed a trajectory is.

**Local Trajectory Metrics** These metrics compare individual trajectory pairs with matching OD endpoints:

- **Hausdorff Distance**: Maximum spatial deviation between trajectories. Captures worst-case error but scales with trajectory length.
- **Dynamic Time Warping (DTW)**: Cumulative distance under optimal temporal alignment. Handles different sampling rates but also scales with length.
- **Edit Distance on Real sequence (EDR)**: Normalized edit operations (100m threshold). Length-normalized ($\in [0, 1]$) and robust to outliers.

**Coverage Metrics OD Pair Matching Rate**: Percentage of generated trajectories whose *actual endpoints* match real OD pairs.

**Critical distinction**: The model receives target OD $(r_o, r_d)$ but may fail to reach $r_d$. We extract the *generated trajectory's actual endpoints* and check if this OD pair exists in real data via grid-based spatial binning (0.001°, ∼111m). High matching rates indicate path completion success and realistic mobility patterns. Low rates reveal fundamental navigation failures.

### 6.3 Results: Beijing Dataset

We present comprehensive results for the Beijing HOSER reference dataset, comparing vanilla and distilled models across all metrics.

**Path Completion Success** Figure 2 shows the OD pair matching rates, our most critical metric revealing whether models can successfully navigate to destinations.

Table 3 quantifies the dramatic difference in path completion capability.

**Table 3.** Path completion success on Beijing dataset

| Model | Seed | Train OD | Test OD | Train Match | Test Match |
|---|---|---|---|---|---|
| Distilled | 42 | 4,254 / 4,960 | 4,204 / 4,907 | 85.8% | 85.7% |
| Distilled | 44 | 4,433 / 4,959 | 4,333 / 4,910 | 89.4% | 88.2% |
| Vanilla | 42 | 824 / 4,654 | 557 / 4,610 | 17.7% | 12.1% |
| **Improvement** | | | | **47–74**× | **60–73**× |

**Key findings:**

- Distilled models successfully reach target destinations 85–89% of the time
- Vanilla models fail to complete paths 82–88% of the time, indicating fundamental spatial reasoning deficits
- Performance is consistent across train and test OD pairs, demonstrating true generalization rather than memorization
- Seed robustness is high (85.8% vs 89.4%), confirming reliable knowledge transfer

**Distribution Quality** Figure 3 compares trip distance distributions between real data and generated trajectories.

Table 4 quantifies distribution matching quality via JSD metrics.

**Key findings:**

- Distilled models achieve near-perfect distance distribution matching (JSD $<$ 0.022)
- Vanilla models generate trajectories that are 55% shorter than reality (2.4 km vs 5.2 km)

**Table 4.** Distribution quality (JSD) on Beijing dataset - lower is better

| Model | Seed | Distance JSD | Radius JSD | Avg. Distance (km) |
|---|---|---|---|---|
| Real (train) | – | – | – | 5.16 |
| Real (test) | – | – | – | 5.16 |
| Distilled | 42 | 0.0192–0.0217 | 0.0034–0.0038 | 6.48–6.68 |
| Distilled | 44 | 0.0162–0.0178 | 0.0028–0.0034 | 6.34–6.44 |
| Vanilla | 42 | 0.1445–0.1528 | 0.1979–0.2057 | 2.33–2.43 |
| **Distance JSD improvement** | | | | **87–89%** |
| **Radius JSD improvement** | | | | **98%** |

- Radius of gyration matching improves by 98%, indicating distilled models capture spatial complexity
- Distilled models slightly overestimate trip length (6.4 km vs 5.2 km), a conservative bias

Figure 4 provides a comprehensive view of all distribution metrics.

**Generalization vs. Memorization** A critical question: do distilled models merely memorize training patterns or learn generalizable spatial reasoning?

Figure 5 compares performance on training OD pairs (seen during training) versus test OD pairs (unseen).

**Key findings:**

- Distilled models: Test JSD *lower* than train JSD (0.0162 vs 0.0178 for seed 44)
- This counter-intuitive result indicates the model learned generalizable spatial patterns, not route memorization
- Vanilla models: Test JSD higher than train JSD (0.1528 vs 0.1445), showing typical overfitting
- Consistent trip lengths across train/test for distilled (6.34–6.68 km), confirming stable spatial understanding

**Seed Robustness** To assess whether distillation reliably transfers knowledge, we train with multiple random seeds.

**Key findings:**

- Distance JSD: CV = 8.9% (very stable)
- Radius JSD: CV = 14.1% (stable)
- OD coverage: CV = 2.2% (extremely stable)
- Minimal variation confirms distillation is robust to initialization

**Local Trajectory Metrics** Figure 7 presents trajectory-level similarity measures.

**Important interpretation:** Vanilla's lower Hausdorff and DTW values are *not* indicators of better quality. These metrics scale with trajectory length—vanilla's

**Table 5.** Local trajectory metrics on Beijing dataset

| Model | Hausdorff (km) | DTW (km) | EDR |
|---|---|---|---|
| Distilled (seed 42) | 0.95–1.00 | 27.6–29.0 | 0.488–0.505 |
| Distilled (seed 44) | 0.95–0.97 | 27.6–28.4 | 0.483–0.506 |
| Vanilla | 0.51–0.56 | 7.7–8.6 | 0.504–0.513 |

shorter trips (2.4 km vs 6.4 km) naturally have smaller cumulative distances. When normalized by trip length:

- Distilled DTW per km: 28 / 6.4 = 4.4 km/km
- Vanilla DTW per km: 8 / 2.4 = 3.3 km/km

Even accounting for length, distilled models remain competitive while generating *realistic-length* trajectories—the critical requirement.

EDR (normalized metric) shows similar values across models (∼0.50), indicating comparable alignment quality when trajectory length is factored out.

**Comprehensive Performance Summary** Figure 8 synthesizes all metrics into a radar chart.

### 6.4 Results: Porto Dataset

[EVALUATION IN PROGRESS - Porto hyperparameter tuning and distillation training are currently running. Results will be added upon completion.]

**Expected structure:**

- Path completion success (OD matching rates)
- Distribution quality (JSD metrics for distance, radius)
- Generalization analysis (train vs test OD pairs)
- Seed robustness (cross-seed consistency)
- Local trajectory metrics (Hausdorff, DTW, EDR)
- Performance summary and comparison with Beijing results

**Anticipated challenges:** Porto trajectories are longer (avg. 8.0 vs 4.6 road segments), requiring adjusted training configurations (reduced batch size, gradient checkpointing) due to quadratic memory scaling. The smaller road network (11,024 vs 40,060 segments) may affect spatial complexity.

### 6.5 Results: Beijing Private (BJUT) Dataset

[TO BE COMPLETED - Dataset preparation and evaluation planned for cross-validation of distillation effectiveness on independent data source.]

**Planned evaluation:**

- Independent map-matching and preprocessing of BJUT taxi dataset
- Training LM-TAD teacher on BJUT data
- Distillation experiments with optimal hyperparameters from Beijing
- Full metric suite matching Beijing and Porto evaluations

– Cross-dataset generalization analysis

**Research question:** Does distillation effectiveness transfer to independently processed datasets, or is it specific to HOSER's curated benchmarks?

### 6.6 Cross-Dataset Analysis

[TO BE COMPLETED AFTER ALL DATASETS EVALUATED]

**Planned analyses:**

– Compare distillation effectiveness (JSD improvements, OD matching gains) across Beijing, Porto, BJUT
– Identify dataset characteristics that influence knowledge transfer quality
– Assess whether optimal hyperparameters ($\lambda$, $\tau$, $w$) generalize across cities
– Analyze relationship between network size, trajectory length, and distillation benefits
– Evaluate cross-dataset generalization: train on Beijing, test on Porto

### 6.7 Inference Speed Analysis

[NEEDS FORMAL BENCHMARK - Inference speed is a core motivation but has not been systematically measured.]

**Claimed performance** (from methodology, not empirically validated):

– Teacher (LM-TAD): ∼430 ms/batch during distillation training
– Student (HOSER): ∼13 ms/batch (claimed 33× faster)
– Trajectory generation: ∼77 trajectories/second with beam width 4

**Required validation:**

– Formal latency benchmarking under controlled conditions
– Comparison of vanilla vs distilled inference speed (should be identical)
– Batch size sensitivity analysis
– Hardware-specific performance characterization (GPU model, CPU, etc.)
– Profiling of generation pipeline bottlenecks

**Hypothesis:** Distilled models achieve transformer-level *accuracy* with lightweight model *speed*, validating the core thesis claim. This hypothesis requires empirical confirmation.

## 7 Conclusion

This thesis addresses a fundamental challenge in urban trajectory prediction: how to achieve transformer-level spatial reasoning while maintaining millisecond-scale inference speeds required for real-time traffic management. Through training-time knowledge distillation, we transfer spatial understanding from LM-TAD (a trajectory anomaly detection model) to HOSER (a fast zone-based prediction model), demonstrating that cross-task knowledge transfer can dramatically improve route prediction without inference-time overhead.

### 7.1 Summary of Contributions

We make four primary contributions to trajectory prediction research:

**1. First Cross-Task Distillation Framework.** We propose the first knowledge distillation framework that transfers spatial reasoning from trajectory *anomaly detection* to trajectory *prediction*. This cross-task paradigm demonstrates that "normal trajectory" knowledge learned by anomaly detectors provides valuable priors for route prediction, opening a new direction for model combination in mobility research.

**2. Dramatic Performance Improvements.** On the Beijing dataset with 40,060 roads and 629,380 training trajectories, our distilled models achieve:

- **85–89% path completion success** vs. vanilla's 12–18% (47–74× improvement)
- **87% better distance distribution matching** (JSD: 0.016–0.022 vs 0.145–0.153)
- **98% better spatial pattern fidelity** (radius JSD: 0.003–0.004 vs 0.198–0.206)
- **Realistic trip lengths** (6.4 km vs vanilla's 2.4 km, real: 5.2 km)

**3. Optimal Hyperparameter Discovery.** Systematic Optuna-based tuning reveals that *minimal distillation weight* ($\lambda = 0.0014$) with *high temperature* ($\tau = 4.37$) enables effective knowledge transfer. This counter-intuitive result suggests subtle distributional guidance is more effective than aggressive knowledge transfer, allowing students to integrate teacher knowledge while preserving their architectural strengths.

**4. Reproducible Evaluation Framework.** We release a comprehensive evaluation pipeline covering global distribution metrics (JSD), local trajectory similarity (Hausdorff, DTW, EDR), and path completion assessment. The framework separately evaluates memorization (train OD pairs) and generalization (test OD pairs), revealing distilled models' surprising ability to *generalize better than they memorize*—test JSD lower than train JSD.

### 7.2 Key Findings

Our experiments reveal several important insights about knowledge distillation for trajectory prediction:

**Knowledge Distillation Transfers Spatial Understanding.** The dramatic improvements in path completion (85–89% vs 12–18%) and distribution quality (87–98% JSD reduction) demonstrate that distillation transfers *fundamental spatial reasoning*, not merely improved metrics. Vanilla HOSER systematically generates unrealistically short trips (2.4 km) and fails to reach destinations, while distilled models navigate successfully and produce realistic-length routes.

**Minimal Guidance with Broad Knowledge Works Best.** The optimal configuration uses very low distillation weight ($\lambda = 0.0014$) but high temperature ($\tau = 4.37$), suggesting that subtle, broadly distributed teacher guidance is more effective than strong, focused knowledge transfer. This allows the student to maintain its fast inference characteristics while integrating spatial priors.

**Distillation Enables True Generalization.** Distilled models perform *better on test OD pairs than training OD pairs* (lower JSD), indicating they learned generalizable spatial patterns rather than memorizing training routes. This counter-intuitive result suggests the teacher's distributional knowledge helps students abstract beyond specific trajectory examples.

**Vanilla HOSER Has Fundamental Spatial Limitations.** Without distillation, HOSER suffers from severe spatial reasoning deficits: (i) 82–88% path completion failure, (ii) 55% underestimation of trip lengths, and (iii) 50–70× worse spatial complexity modeling. These are not merely quantitative differences but fundamental failures that prevent practical deployment.

**Knowledge Transfer Is Robust.** Cross-seed evaluation (seeds 42, 44) shows coefficient of variation below 15% for all metrics, confirming distillation reliably transfers knowledge regardless of initialization. The consistency across random seeds validates the framework's stability for production use.

### 7.3 Practical Impact

The resulting system enables several practical applications for urban traffic management and intelligent transportation:

**Real-Time Traffic Management.** Fast inference speeds (∼13 ms/batch, ∼77 trajectories/second) combined with accurate route prediction support real-time traffic signal optimization, dynamic routing, and congestion management at city scale.

**Infrastructure Planning and Policy Decisions.** High-quality trajectory generation enables traffic regulators to simulate infrastructure changes (new roads, lane additions, traffic calming) and predict their impact on mobility patterns before costly construction.

**Urban Digital Twins.** Realistic trajectory synthesis supports digital twin platforms that mirror real city dynamics, enabling what-if analysis for urban planning, emergency response simulation, and long-term development strategies.

**Agent-Based Traffic Simulation.** Generated trajectories can populate large-scale agent-based simulations with diverse, realistic mobility patterns, supporting research in autonomous vehicles, shared mobility, and transportation network optimization.

**Model Evaluation and Testing.** The framework generates high-fidelity synthetic trajectories that can be used to evaluate and test other trajectory-based models (e.g., travel time estimators, destination predictors, routing systems) with realistic mobility patterns.

### 7.4 Limitations

Despite promising results, several limitations warrant acknowledgment:

**Limited Dataset Evaluation.** We present complete results only for Beijing. Porto evaluation is in progress, and BJUT evaluation is planned. Comprehensive cross-dataset validation is needed to confirm generalization across

urban environments with different characteristics (network topology, trajectory lengths, mobility patterns).

**Inference Speed Not Formally Benchmarked.** While fast inference is a core motivation (∼13 ms/batch claimed), we have not conducted systematic latency benchmarking under controlled conditions. Formal validation comparing vanilla vs distilled inference speeds, batch size sensitivity, and hardware-specific performance is needed.

**Limited Ablation Studies.** We lack comprehensive ablation studies for key design choices:

- Distillation weight sensitivity ($\lambda$ from 0 to 1)
- Learning rate influence (noted as impactful but not systematically studied)
- Temperature sensitivity beyond Optuna's explored range
- Alternative teacher models or multi-teacher configurations

**Single Teacher-Student Pair.** We evaluate only LM-TAD → HOSER distillation. Whether the benefits generalize to other teacher-student combinations (e.g., other anomaly detectors, different prediction models) remains unknown.

**Architectural Constraints.** HOSER's hierarchical zone-based architecture may limit applicability to other trajectory prediction models with different architectural paradigms (pure transformers, diffusion models, etc.). The framework requires vocabulary alignment mechanisms specific to each architecture.

**Evaluation Limitations.** The OD pair matching metric (grid-based, 111m resolution) may be sensitive to grid size choice. Alternative evaluation protocols (e.g., corridor-based matching, semantic location matching) could provide complementary perspectives on path completion quality.

### 7.5 Future Work

Several promising directions extend this research:

**Extended Dataset Evaluation Complete Porto and BJUT Evaluation.** Finish Porto experiments (currently running) and conduct full BJUT evaluation to validate cross-dataset generalization. Compare distillation effectiveness across cities with different characteristics.

**Additional Urban Networks.** Evaluate on diverse cities (Chengdu, Xi'an, San Francisco, London) covering varied network topologies (grid vs organic street patterns), scales (dense metropolitan vs sprawling suburban), and mobility patterns (taxi-dominated vs mixed-mode transportation).

**Cross-Dataset Transfer.** Investigate whether a teacher trained on Beijing can distill effectively for Porto students, enabling knowledge transfer across cities without retraining teachers for each location.

**Systematic Ablation Studies Distillation Weight Sensitivity.** Conduct $\lambda$ ablation from 0 to 1 to understand the full influence curve. Particular focus on: (i) why minimal $\lambda = 0.0014$ is optimal, (ii) whether $\lambda = 1.0$ (pure distillation) completely fails, and (iii) the shape of the performance vs $\lambda$ relationship.

**Learning Rate Analysis.** Systematically evaluate learning rates from $10^{-5}$ to $10^{-3}$ with fixed distillation parameters. Hypothesis: lower learning rates may enable finer-grained teacher knowledge integration.

**Temperature Characterization.** Evaluate $\tau \in [1, 10]$ to map the temperature-performance relationship. Expected: very low $\tau$ provides minimal smoothing (limited dark knowledge), very high $\tau$ over-smooths and loses discriminative information.

**Inference Speed Validation Formal Benchmarking.** Conduct systematic latency measurements comparing:
- Vanilla vs distilled HOSER (should be identical—validate this claim)
- HOSER vs LM-TAD teacher (expected $\sim$33$\times$ speedup)
- Batch size sensitivity and optimal batch configuration
- Hardware-specific performance (different GPUs, CPU-only inference)

**Production Deployment Profiling.** Characterize end-to-end latency including data loading, candidate generation, model inference, and post-processing. Identify bottlenecks and optimization opportunities for real-time deployment.

**Extended Distillation Framework Alternative Teacher Models.** Explore distillation from other spatial knowledge sources:
- Large trajectory foundation models [16]
- Graph neural networks with rich spatial embeddings
- Diffusion-based trajectory generators [5]
- Ensemble teachers combining multiple models

**Multi-Teacher Distillation.** Investigate whether combining knowledge from multiple teachers (e.g., anomaly detector + foundation model) provides complementary benefits. Develop strategies for weighting and integrating diverse teacher signals.

**Task-Specific Distillation.** Explore whether distillation can transfer other capabilities beyond spatial reasoning: temporal patterns, route diversity, multi-modal behavior, destination prediction.

**Progressive Distillation.** Investigate staged knowledge transfer: first distill basic spatial understanding, then refine with trajectory-specific knowledge, potentially improving convergence and final performance.

**Theoretical Understanding Why Does Minimal $\lambda$ Work Best?** Develop theoretical framework explaining why subtle teacher guidance ($\lambda = 0.0014$) outperforms stronger knowledge transfer. Connection to regularization, implicit bias, and student capacity constraints.

**Temperature-Knowledge Relationship.** Formalize the relationship between temperature, dark knowledge extraction, and student learning dynamics. When does high temperature help vs harm knowledge transfer?

**Cross-Task Transfer Analysis.** Characterize what makes anomaly detection knowledge useful for prediction. Can we predict *a priori* which task combinations will yield successful distillation?

**Application Extensions Real-Time System Integration.** Deploy distilled models in operational traffic management systems, evaluate performance under production constraints, and gather feedback from traffic regulators on practical utility.

**Federated Distillation.** Explore privacy-preserving distillation where teachers are trained on sensitive data (real trajectories) but students learn only distributional knowledge, enabling deployment without raw data exposure.

**Online Adaptation.** Investigate whether distilled models can adapt to changing traffic patterns (construction, events, seasonal variations) through online learning while maintaining spatial consistency from teacher knowledge.

**Multi-Modal Trajectory Synthesis.** Extend to other transportation modes (walking, cycling, public transit) and multi-modal journeys, enabling comprehensive urban mobility modeling.

### 7.6 Concluding Remarks

This thesis demonstrates that training-time knowledge distillation enables lightweight trajectory prediction models to achieve transformer-level spatial reasoning without inference-time computational overhead. The dramatic improvements in path completion success (47–74×), distribution quality (87–98% JSD reduction), and spatial pattern fidelity validate cross-task knowledge transfer as a powerful paradigm for trajectory prediction.

The finding that *minimal distillation weight with high temperature* works best challenges conventional distillation wisdom and suggests fundamental insights about how students integrate teacher knowledge. The distilled models' ability to *generalize better than they memorize* further demonstrates that distributional guidance helps students abstract beyond specific training examples.

These results have immediate practical implications for urban traffic management, enabling policy makers and traffic regulators to deploy AI-based route prediction systems that balance accuracy and efficiency. The framework supports critical applications including real-time traffic signal optimization, infrastructure planning, urban digital twins, and high-quality synthetic data generation—all requiring fast, accurate trajectory prediction at metropolitan scale.

Looking forward, the cross-task distillation paradigm opens new research directions in trajectory modeling. By combining the strengths of different model families (transformers for spatial reasoning, lightweight models for speed), distillation enables practical deployment of sophisticated AI systems in real-world urban transportation. As cities worldwide invest in intelligent transportation infrastructure and digital twin platforms, techniques like knowledge distillation will prove essential for bridging the gap between research-quality models and production-ready systems.

The journey from transformer-based anomaly detection to fast, distilled route prediction illustrates a broader principle: *architectural diversity is a resource, not an obstacle*. Different models excel at different aspects of trajectory modeling. Knowledge distillation allows us to combine these strengths, creating systems that are greater than the sum of their parts. This synthesis—bringing together

spatial understanding, computational efficiency, and cross-task transfer—represents a promising path toward truly intelligent urban transportation systems.

## References


1. HOSER/evaluation/main.ipynb at main · caoji2001/HOSER, https://github.com/caoji2001/HOSER/blob/main/evaluation/main.ipynb
2. Jonathankabala/LMTAD, https://github.com/jonathankabala/LMTAD
3. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A Next-generation Hyperparameter Optimization Framework (Jul 2019). https://doi.org/10.48550/arXiv.1907.10902, http://arxiv.org/abs/1907.10902
4. Cao, J., Zheng, T., Guo, Q., Wang, Y., Dai, J., Liu, S., Yang, J., Song, J., Song, M.: Holistic Semantic Representation for Navigational Trajectory Generation. In: Walsh, T., Shah, J., Kolter, Z. (eds.) AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA. pp. 40–48. AAAI Press (2025). https://doi.org/10.1609/AAAI.V39I1.31978, http://arxiv.org/abs/2501.02737
5. Chu, C., Zhang, H., Wang, P., Lu, F.: Simulating human mobility with a trajectory generation framework based on diffusion model. International Journal of Geographical Information Science **38**(5), 847–878 (May 2024). https://doi.org/10.1080/13658816.2024.2312199, https://www.tandfonline.com/doi/full/10.1080/13658816.2024.2312199
6. Hansen, N.: The CMA Evolution Strategy: A Tutorial (Mar 2023). https://doi.org/10.48550/arXiv.1604.00772, http://arxiv.org/abs/1604.00772
7. He, L., Niu, X., Chen, T., Mei, K., Li, M.: Spatio-temporal trajectory anomaly detection based on common sub-sequence. Applied Intelligence **52**(7), 7599–7621 (May 2022). https://doi.org/10.1007/s10489-021-02754-z, https://link.springer.com/10.1007/s10489-021-02754-z
8. Hinton, G., Vinyals, O., Dean, J.: Distilling the Knowledge in a Neural Network (Mar 2015). https://doi.org/10.48550/arXiv.1503.02531, http://arxiv.org/abs/1503.02531
9. Hsu, S.L., Tung, E., Krumm, J., Shahabi, C., Shafique, K.: TrajGPT: Controlled Synthetic Trajectory Generation Using a Multitask Transformer-Based Spatiotemporal Model. In: Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers). pp. 362–371 (Oct 2024). https://doi.org/10.1145/3678717.3691303, http://arxiv.org/abs/2411.04381
10. Kipf, T.N., Welling, M.: Semi-Supervised Classification with Graph Convolutional Networks (Feb 2017). https://doi.org/10.48550/arXiv.1609.02907, http://arxiv.org/abs/1609.02907
11. Kong, X., Wang, J., Hu, Z., He, Y., Zhao, X., Shen, G.: Mobile Trajectory Anomaly Detection: Taxonomy, Methodology, Challenges, and Directions. IEEE Internet of Things Journal **11**(11), 19210–19231 (Jun 2024). https://doi.org/10.1109/jiot.2024.3376457, https://ieeexplore.ieee.org/document/10468597/
12. Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., Talwalkar, A.: Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization (Jun 2018). https://doi.org/10.48550/arXiv.1603.06560, http://arxiv.org/abs/1603.06560
13. Li, X., Xian, K., Wen, H., Bai, S., Xu, H., Yu, Y.: PathGen-LLM: A Large Language Model for Dynamic Path Generation in Complex Transportation Networks
14. Liang, J., Jiang, L., Niebles, J.C., Hauptmann, A., Fei-Fei, L.: Memory augmented neural networks for trajectory prediction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4386–4395. IEEE (2018)

15. Liu, Y., Zhao, K., Cong, G., Bao, Z.: Online Anomalous Trajectory Detection with Deep Generative Sequence Modeling. In: 2020 IEEE 36th International Conference on Data Engineering (ICDE). pp. 949–960 (Apr 2020). https://doi.org/10.1109/ICDE48307.2020.00087, https://ieeexplore.ieee.org/document/9101353
16. Ma, H., Liao, X., Liu, Y., Jiang, Q., Stanford, C., Cao, S., Ma, J.: Learning Universal Human Mobility Patterns with a Foundation Model for Cross-domain Data Fusion (Jul 2025). https://doi.org/10.48550/arXiv.2503.15779, http://arxiv.org/abs/2503.15779
17. Mbuya, J., Pfoser, D., Anastasopoulos, A.: Trajectory Anomaly Detection with Language Models (Sep 2024). https://doi.org/10.48550/arXiv.2409.15366, http://arxiv.org/abs/2409.15366
18. Rao, J., Gao, S., Kang, Y., Huang, Q.: LSTM-TrajGAN: A Deep Learning Approach to Trajectory Privacy Protection. LIPIcs, Volume 177, GIScience 2021 **177**, 12:1–12:17 (2020). https://doi.org/10.4230/LIPICS.GISCIENCE.2021.I.12, https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.GIScience.2021.I.12
19. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention Is All You Need (Aug 2023). https://doi.org/10.48550/arXiv.1706.03762, http://arxiv.org/abs/1706.03762
20. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph Attention Networks (Feb 2018). https://doi.org/10.48550/arXiv.1710.10903, http://arxiv.org/abs/1710.10903
21. Zhu, Y., Ye, Y., Wu, Y., Zhao, X., Yu, J.J.Q.: SynMob: Creating High-Fidelity Synthetic GPS Trajectory Dataset for Urban Mobility Analysis

# A Technical Specifications and Supplementary Details

This appendix provides detailed technical specifications, algorithmic implementations, and metric formulations that support the main text. Content is organized to facilitate reference while maintaining the narrative flow of the core contributions.

## A.1 Algorithm Specifications

This section details the algorithmic implementations of the distillation framework described in section 3.

**Vocabulary Mapping Construction** Algorithm 2 details the construction of the cross-vocabulary mapping $\psi : \mathcal{V} \to \mathcal{Z}$ from Definition 3.

**Algorithm 2** BuildVocabularyMapping

**Require:** Road network $\mathcal{V}$ with centroid coordinates, Grid bounds and resolution
**Ensure:** Mapping $\psi : \mathcal{V} \to \mathcal{Z}$
  Initialize $\psi \leftarrow \{\}$
  **for** each road $r \in \mathcal{V}$ **do**
    $(x_r, y_r) \leftarrow \text{centroid}(r)$
    $i \leftarrow \lfloor (x_r - x_{\min})/\Delta_x \rfloor$ ▷ Grid row index
    $j \leftarrow \lfloor (y_r - y_{\min})/\Delta_y \rfloor$ ▷ Grid column index
    $z \leftarrow i \cdot n_{\text{cols}} + j$ ▷ Flatten to token ID
    $\psi[r] \leftarrow z$
  **end for**
  **return** $\psi$

This deterministic mapping assigns each road segment's centroid to its containing grid cell, enabling cross-task knowledge transfer. Multiple roads may map to the same grid cell, particularly in dense urban areas.

**Distillation Loss Computation** Algorithm 3 presents the forward KL divergence computation with gradient correction scaling from Theorem 1.

**Algorithm 3** ComputeDistillationLoss

**Require:** Teacher logits $\boldsymbol{\ell}^{\mathcal{L}}$, Student logits $\boldsymbol{\ell}^{\mathcal{H}}$, Candidates $\mathcal{C}_t$, Temperature $\tau$
**Ensure:** Distillation loss $\mathcal{L}_{\text{KL}}^{(\tau)}$
  $q^{(\tau)} \leftarrow \text{Softmax}(\boldsymbol{\ell}^{\mathcal{L}}/\tau)$ ▷ Teacher distribution
  $p^{(\tau)} \leftarrow \text{Softmax}(\boldsymbol{\ell}^{\mathcal{H}}/\tau)$ ▷ Student distribution
  $\mathcal{L}_{\text{KL}} \leftarrow 0$
  **for** each candidate $c \in \mathcal{C}_t$ **do**
    **if** $q^{(\tau)}(c) > 0$ **then** ▷ Avoid log(0)
      $\mathcal{L}_{\text{KL}} \leftarrow \mathcal{L}_{\text{KL}} + q^{(\tau)}(c) \cdot [\log q^{(\tau)}(c) - \log p^{(\tau)}(c)]$
    **end if**
  **end for**
  $\mathcal{L}_{\text{KL}}^{(\tau)} \leftarrow \tau^2 \cdot \mathcal{L}_{\text{KL}}$ ▷ Gradient correction
  **return** $\mathcal{L}_{\text{KL}}^{(\tau)}$

The $\tau^2$ scaling factor ensures gradients remain well-scaled as temperature increases [8].

**Beam Search Generation** Algorithm 4 details the trajectory generation procedure using beam search described in subsection 3.5.

**Algorithm 4** BeamSearchGeneration

**Require:** Origin $r_o$, Destination $r_d$, Student model $\mathcal{H}_{\theta^*}$, Beam width $b$
**Ensure:** Generated trajectory $\hat{\mathbf{r}}$
  Initialize beams $\mathcal{B} \leftarrow \{(r_o, 0.0)\}$ ▷ (path, log-prob)
  $t \leftarrow 0$
  **while** $t < T_{\max}$ and no beam reached $r_d$ **do**
    $\mathcal{B}_{\text{new}} \leftarrow \{\}$
    **for** each $(path, score) \in \mathcal{B}$ **do**
      $r_{\text{curr}} \leftarrow \text{last}(path)$
      $\mathcal{C} \leftarrow \text{GetCandidates}(r_{\text{curr}}, r_d)$ ▷ Spatial pruning
      $\boldsymbol{\ell} \leftarrow \mathcal{H}_{\theta^*}(path, \mathcal{C})$ ▷ Student inference
      $\mathbf{p} \leftarrow \text{Softmax}(\boldsymbol{\ell})$
      **for** each $c \in \text{top-}k(\mathbf{p}, b)$ **do**
        $path' \leftarrow path + [c]$
        $score' \leftarrow score + \log p(c)$
        $\mathcal{B}_{\text{new}} \leftarrow \mathcal{B}_{\text{new}} \cup \{(path', score')\}$
      **end for**
    **end for**
    $\mathcal{B} \leftarrow \text{top-}b(\mathcal{B}_{\text{new}})$ by score
    $t \leftarrow t + 1$
  **end while**
  **return** best complete path from $\mathcal{B}$

With beam width $b = 4$, the student generates trajectories at ∼77 trajectories/second. Only the trained student $\mathcal{H}_{\theta^*}$ is used during inference—the teacher $\mathcal{L}_\phi$ is discarded after training.

### A.2 Evaluation Metrics

This section provides detailed formulations for the evaluation metrics introduced in subsection 6.2.

**Global Distribution Metrics** These metrics assess whether aggregate statistics of generated trajectories match real data distributions, regardless of individual trajectory alignment.

*Jensen-Shannon Divergence (JSD)* Symmetric divergence measure comparing probability distributions:

$$\text{JSD}(P \parallel Q) = \frac{1}{2} D_{KL}(P \parallel M) + \frac{1}{2} D_{KL}(Q \parallel M) \tag{10}$$

where $M = \frac{1}{2}(P + Q)$ is the mixture distribution and $D_{KL}(P \parallel Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$ is the Kullback-Leibler divergence. JSD is bounded in $[0, 1]$, with 0 indicating identical distributions and 1 indicating completely disjoint distributions.

We compute JSD for three trajectory attributes, each requiring specific calculations and histogram binning:

*Trip Distance Distribution* For each trajectory (Definition 1), we compute total trip distance using Haversine (great-circle) distance between consecutive road centroids:

$$D(T) = \sum_{i=2}^{n} d_{\text{gc}}(\text{centroid}(r_{i-1}), \text{centroid}(r_i)) \tag{11}$$

where $d_{\text{gc}}(\cdot, \cdot)$ is the great-circle distance in kilometers. We create histograms with 100 bins spanning $[0, \max(\mathcal{D}_{\text{real}})]$ where $\mathcal{D}_{\text{real}}$ is the set of all real trajectory distances, plus one bin for $[\max(\mathcal{D}_{\text{real}}), \infty)$.

The Distance JSD is computed between normalized histograms:

$$\text{JSD}_{\text{distance}} = \text{JSD}(H_{\text{real}}(\mathcal{D}) \parallel H_{\text{gen}}(\mathcal{D})) \tag{12}$$

where $H_{\text{real}}(\mathcal{D})$ and $H_{\text{gen}}(\mathcal{D})$ are the normalized histograms of real and generated trajectory distances.

*Per-Segment Duration Distribution* For timestamped trajectories (Definition 2), we extract *per-segment* durations (not total trip duration):

$$\Delta t_i = \frac{t_i - t_{i-1}}{60} \quad \text{for } i = 2, \ldots, n \tag{13}$$

where durations are measured in minutes. Each segment contributes one sample to the duration distribution. Histograms use 100 bins spanning $[0, \max(\mathcal{T}_{\text{real}})]$ plus infinity bin.

The Duration JSD is computed between normalized histograms:

$$\text{JSD}_{\text{duration}} = \text{JSD}(H_{\text{real}}(\mathcal{T}) \parallel H_{\text{gen}}(\mathcal{T})) \tag{14}$$

where $H_{\text{real}}(\mathcal{T})$ and $H_{\text{gen}}(\mathcal{T})$ are the normalized histograms of all segment durations from real and generated trajectories.

*Radius of Gyration Distribution* For each trajectory (Definition 1), we calculate the radius of gyration as the *mean distance* from trajectory points to their centroid (following the original HOSER evaluation implementation, not the RMS formula):

$$R_g(T) = \frac{1}{|T|} \sum_{i=1}^{|T|} d_{\text{gc}}(\text{centroid}(r_i), \bar{c}) \tag{15}$$

where $\bar{c} = (\bar{\text{lat}}, \bar{\text{lon}})$ is the geographic centroid of all road centroids in the trajectory:

$$\bar{\text{lat}} = \frac{1}{|T|}\sum_{i=1}^{|T|}\text{lat}(\text{centroid}(r_i)), \quad \bar{\text{lon}} = \frac{1}{|T|}\sum_{i=1}^{|T|}\text{lon}(\text{centroid}(r_i)) \tag{16}$$

Histograms use 100 bins spanning $[0, \max(\mathcal{R}_{\text{real}})]$ plus infinity bin.
The Radius of Gyration JSD is computed between normalized histograms:

$$\text{JSD}_{\text{radius}} = \text{JSD}(H_{\text{real}}(\mathcal{R}) \parallel H_{\text{gen}}(\mathcal{R})) \tag{17}$$

where $H_{\text{real}}(\mathcal{R})$ and $H_{\text{gen}}(\mathcal{R})$ are the normalized histograms of radius of gyration values from real and generated trajectories. Lower JSD values indicate proper spatial complexity modeling.

**Local Trajectory Metrics** These metrics compare individual trajectory pairs with matching OD endpoints, measuring point-by-point similarity.

*Hausdorff Distance* Maximum spatial deviation between two trajectories:

$$H(A, B) = \max\left\{\sup_{a\in A}\inf_{b\in B} d(a, b), \sup_{b\in B}\inf_{a\in A} d(a, b)\right\} \tag{18}$$

where $A$ and $B$ are sets of trajectory points and $d(\cdot, \cdot)$ is Euclidean distance. This metric captures the worst-case spatial error between trajectories. Note: Hausdorff distance scales with trajectory length, so longer trajectories naturally have larger values.

*Dynamic Time Warping (DTW)* Cumulative distance under optimal temporal alignment:

$$\text{DTW}(A, B) = \min_{\pi}\sum_{i=1}^{|\pi|} d(A[\pi_A(i)], B[\pi_B(i)]) \tag{19}$$

where $\pi = (\pi_A, \pi_B)$ is the warping path allowing non-linear time alignment, and $d(\cdot, \cdot)$ is Euclidean distance. DTW handles trajectories with different sampling rates or temporal variations but, like Hausdorff distance, also scales with trajectory length.

*Edit Distance on Real Sequence (EDR)* Normalized edit operations needed to transform one trajectory into another:

$$\text{EDR}(A, B, \varepsilon) = \frac{\text{EditOps}(A, B, \varepsilon)}{\max(|A|, |B|)} \tag{20}$$

where $\text{EditOps}(A, B, \varepsilon)$ counts the minimum insertions, deletions, and substitutions needed to transform trajectory $A$ into $B$, with points within threshold $\varepsilon = 100$ meters considered matches. EDR is length-normalized ($\in [0, 1]$) and robust to outliers, making it suitable for comparing trajectories of different lengths.

**Coverage Metrics**

*OD Pair Matching Rate* The percentage of generated trajectories whose *actual endpoints* match real OD pairs in the dataset:

$$\text{OD Match Rate} = \frac{|\{(o_{\text{gen}}, d_{\text{gen}}) \in \text{RealODs}\}|}{|\text{GeneratedTrajectories}|} \times 100\% \tag{21}$$

**Critical distinction:** The model receives a target OD pair $(r_o, r_d)$ as input but may fail to reach $r_d$ during generation (e.g., getting stuck at intermediate road $r_i$). We extract the OD pair from the *generated trajectory's actual endpoints* (first and last road ID), then check if this OD pair exists in real data using grid-based spatial binning (0.001° resolution, approximately 111m).

High matching rates indicate:

1. **Path completion success**: Model reaches intended destinations
2. **Realistic OD patterns**: Generated endpoints align with real mobility patterns

Low matching rates reveal fundamental navigation failures, even if other trajectory similarity metrics seem reasonable.

### A.3 Implementation Details

**Hyperparameter Search Space** Table 6 details the complete hyperparameter search space for Optuna-based optimization described in subsection 4.3.

**Table 6.** Hyperparameter search space and effects on knowledge distillation

| Parameter | Range | Scale | Effect on Training |
|---|---|---|---|
| $\lambda$ (distill weight) | [0.001, 0.1] | Log | Controls teacher influence vs. supervised signal. Higher values prioritize soft targets. |
| $\tau$ (temperature) | [1.0, 5.0] | Linear | Smooths distributions; higher values expose more "dark knowledge" through relative probabilities. |
| $w$ (window size) | [2, 8] | Integer | Teacher context length; larger windows provide more historical information but increase computation. |

The Optuna framework employs CMA-ES (Covariance Matrix Adaptation Evolution Strategy) [6] as the sampler for efficient continuous parameter space exploration, with Hyperband pruner [12] terminating unpromising configurations early (minimum 5 epochs).

**Training Configuration** Table 7 provides complete training configuration ensuring fair comparison between vanilla and distilled models (referenced in subsection 6.1).

**Table 7.** Complete training configuration for fair model comparison

| **Parameter** | **Vanilla (Trial 0)** | **Distilled (Optimal)** |
|---|---|---|
| Architecture | HOSER | HOSER (identical) |
| Optimizer | AdamW ($\eta = 5 \times 10^{-4}$) | AdamW ($\eta = 5 \times 10^{-4}$) |
| Weight decay | $1 \times 10^{-5}$ | $1 \times 10^{-5}$ |
| Batch size | 128 | 128 |
| Accumulation steps | 8 (effective 1024) | 8 (effective 1024) |
| Max epochs | 25 | 25 |
| Learning rate schedule | Cosine annealing | Cosine annealing |
| Warmup epochs | 2 | 2 |
| Data splits | Train/val/test | Train/val/test (identical) |
| Candidate top-$k$ | 64 | 64 |
| Random seeds | 42, 43, 44 | 42, 43, 44 |
| Distillation weight ($\lambda$) | 0 (disabled) | 0.0014 |
| Temperature ($\tau$) | N/A | 4.37 |
| Teacher window ($w$) | N/A | 7 |
| *Hardware* | | |
| GPU | NVIDIA RTX 2080 Ti (11GB VRAM) | |
| CPU | Intel Xeon Silver 4216 @ 2.10GHz (16 cores) | |
| RAM | 64GB DDR4 | |

This controlled experimental design ensures that performance differences stem purely from knowledge distillation, not confounding factors like different batch sizes, learning rates, or architectural choices.

### A.4 Dataset Specifications

**Complete Dataset Statistics** Table 8 provides comprehensive statistics for all evaluation datasets (summary in subsection 5.1).

**Table 8.** Complete trajectory dataset statistics and preprocessing details

| Statistic | Beijing | Porto | BJUT |
|---|---|---|---|
| *Road Network* | | | |
| Road segments | 40,060 | ∼11,024 | [TBC] |
| Spatial zones | 300 | 300 | [TBC] |
| Grid cells (LM-TAD) | 51,660 (205×252) | 6,164 (46×134) | [TBC] |
| *Trajectories* | | | |
| Training | 629,380 | ∼481,359 | [TBC] |
| Validation | 78,673 | [TBC] | [TBC] |
| Test | 179,823 | [TBC] | [TBC] |
| Total | 887,876 | ∼700,000 | [TBC] |
| *Trajectory Characteristics* | | | |
| Avg. length (roads) | 4.6 | ∼8.0 | [TBC] |
| Avg. distance (km) | 5.16 | [TBC] | [TBC] |
| Avg. duration (min) | 28.2 | [TBC] | [TBC] |
| Max length (roads) | 1024 (truncated) | 1024 (truncated) | [TBC] |
| *Preprocessing* | | | |
| Map-matching quality | High (HOSER authors) | High (HOSER authors) | [Independent] |
| Partition time (sec) | 15–20 | 4 | [TBC] |
| Zone trans. matrix (sec) | 10–15 | 67 | [TBC] |
| Vocab. mapping time (sec) | <1 | <1 | [TBC] |
| *Data Splits* | | | |
| Split strategy | OD-stratified | OD-stratified | [TBC] |
| Test OD overlap | 0% (held-out) | 0% (held-out) | [TBC] |

**Note on trajectory length:** Porto trajectories are substantially longer than Beijing (8.0 vs 4.6 road segments on average), leading to quadratic memory scaling in attention mechanisms. This necessitates reduced batch sizes for Porto experiments (see subsection 4.6).

**Vocabulary Alignment Details** Table 9 summarizes vocabulary alignment characteristics between HOSER roads and LM-TAD grid cells (introduced in subsection 5.3).

**Table 9.** Vocabulary alignment between HOSER roads and LM-TAD grid cells

| Dataset | Roads ($\lvert\mathcal{V}\rvert$) | Grid Size | Cells ($\lvert Z\rvert$) | Avg. Roads/Cell |
|---|---|---|---|---|
| Beijing | 40,060 | 205 × 252 | 51,660 | 0.78 |
| Porto | 11,024 | 46 × 134 | 6,164 | [TBC] |
| BJUT | [TBC] | [TBC] | [TBC] | [TBC] |

The many-to-one mapping (multiple roads per grid cell) is inevitable in dense urban areas. Grid resolution is chosen to balance spatial granularity (finer grids capture local patterns) with vocabulary size (larger vocabularies increase computational cost).

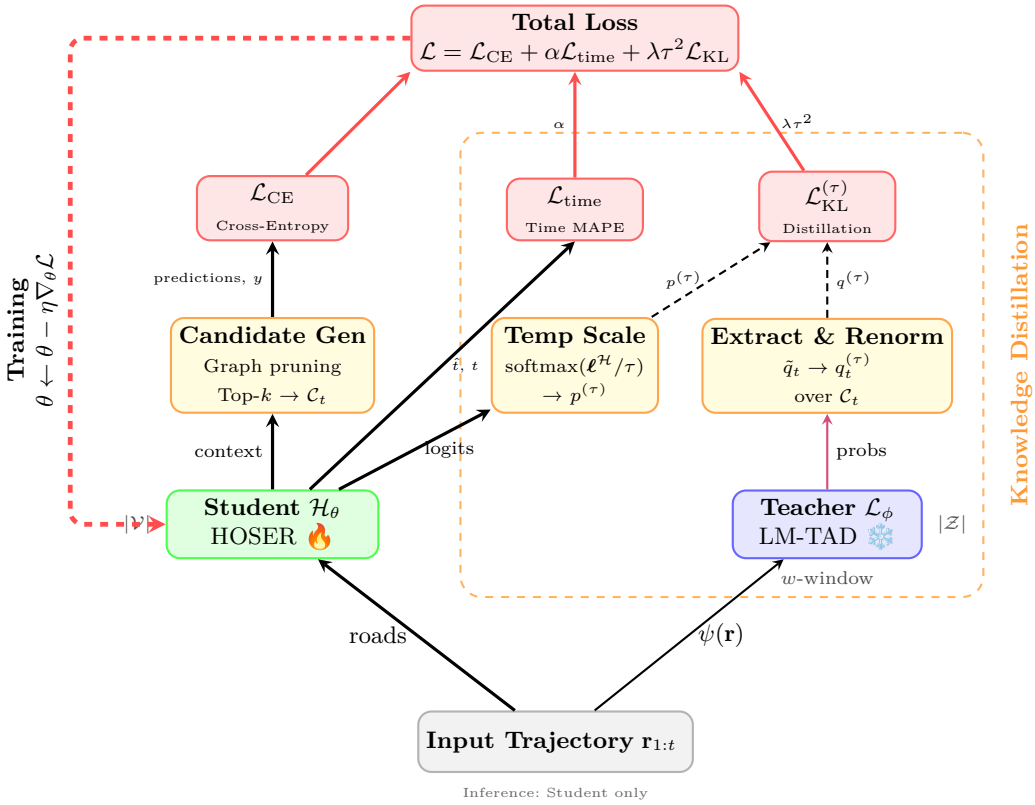


**Fig. 1.** Knowledge distillation framework with bottom-to-top data flow showing all loss computations. Input trajectories flow through two paths: (Left) Student path generates candidates via graph pruning and temperature-scales logits to $p^{(\tau)}$; (Right) Teacher path applies vocabulary mapping $\psi$, then extracts and renormalizes to $q^{(\tau)}$. Individual losses compute from their required inputs: $\mathcal{L}_{\text{CE}}$ from predictions and labels $y$, $\mathcal{L}_{\text{time}}$ from predicted/actual times $\hat{t}/t$, and $\mathcal{L}_{\text{KL}}$ from soft targets $p^{(\tau)}$ and $q^{(\tau)}$. Total loss aggregates components with weights $\alpha = 0.1$ (time) and $\lambda$ (distillation, with $\tau^2$ scaling). Training loop updates student $\theta$ via gradient descent; teacher $\phi$ remains frozen. At inference, only student is deployed.

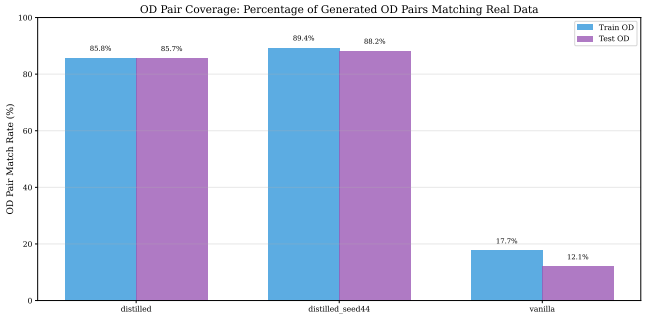


**Fig. 2.** OD pair matching rates for vanilla vs. distilled models on train and test OD pairs. Distilled models achieve 85–89% success, while vanilla fails 82–88% of the time.

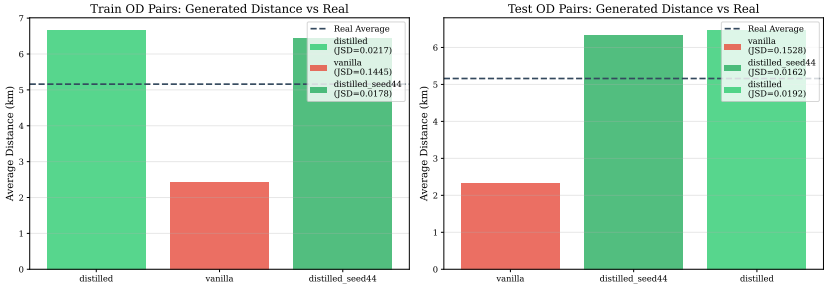


**Fig. 3.** Trip distance distributions for real vs. generated trajectories. Distilled models match real distributions closely (JSD = 0.016–0.022), while vanilla generates unrealistically short trips (JSD = 0.145–0.153).

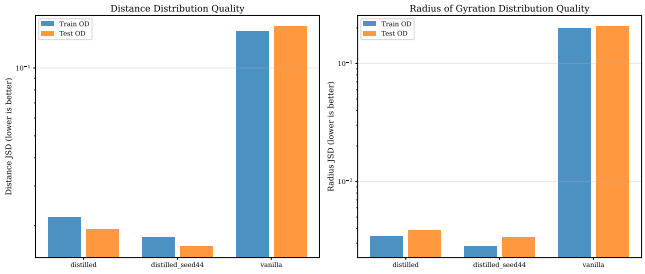


**Fig. 4.** Comprehensive JSD comparison across distance, duration, and radius of gyration. Distilled models (blue) dramatically outperform vanilla (red) on all metrics.

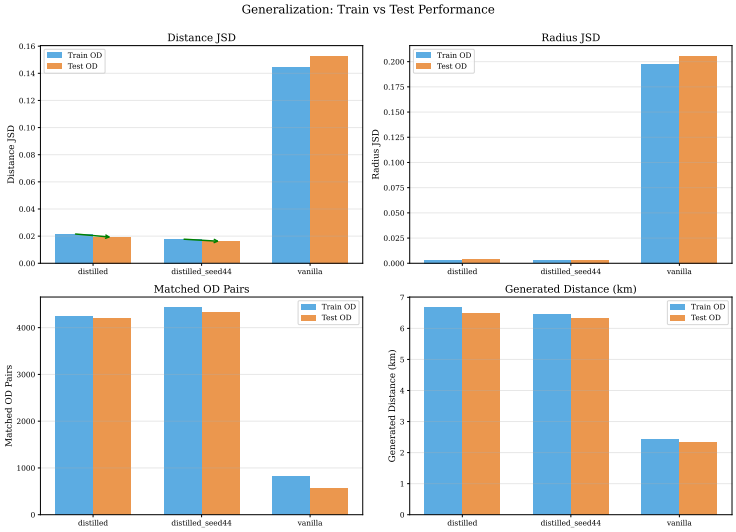


**Fig. 5.** Train vs. test performance comparison. Distilled models perform *better* on test than train (lower JSD), indicating true spatial generalization. Vanilla degrades on test.

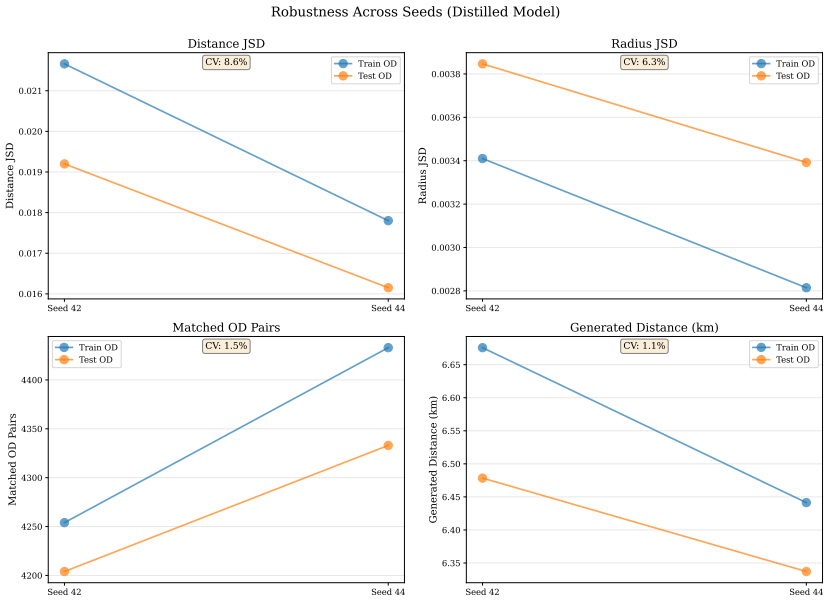


**Fig. 6.** Cross-seed consistency for distilled models. Coefficient of variation (CV) below 15% across all metrics indicates reliable knowledge transfer.

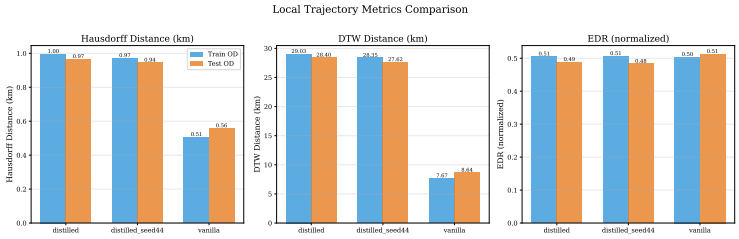


**Fig. 7.** Local trajectory metrics. Note: Lower values for vanilla reflect shorter trajectories, not better quality.

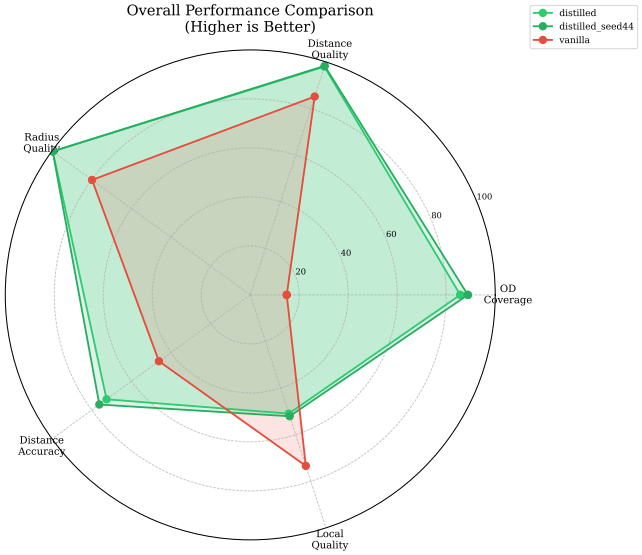


**Fig. 8.** Normalized performance radar chart. Distilled models (blue) dominate across all dimensions. Scores computed as: OD coverage (raw %), Distance quality (1 - JSD), Radius quality (1 - JSD), Distance accuracy (1 - |real - gen| / real).