Multi-Source Urban Traffic Prediction using Drone and Loop Detector Data

Weijiang Xiong[1*], Robert Fonod[1], and Nikolas Geroliminis[1]

[1]École Polytechnique Fédérale de Lausanne (EPFL), Switzerland

Short summary

Traffic forecasting has been a fundamental task in transportation research, with many methods and datasets mainly based on highway loop detector data. In recent years, drones are becoming a favorable choice for urban traffic monitoring, due to their flexibility, high data quality and larger spatial coverage. However, the lack of public datasets has made the joint use of drone and loop detector data fairly under-explored. Therefore, we create a novel simulated multi-source dataset SimBarca for urban traffic prediction, featuring speed measurements from both drones and loop detectors. We provide a graph-based baseline model HiMSNet to handle multiple input modalities and evaluate it along with two benchmark predictors. Our analysis shows that HiMSNet achieves good performance for regional speed prediction, but the road segment-level prediction still requires more in-depth efforts.

Keywords: Traffic forecasting, Drone data, Sensor fusion, Graph Neural Networks, Deep Learning

1 Introduction

Knowing the current traffic states and their future evolution is essential for traffic management and control in Intelligent Transportation Systems (ITS). Therefore, predicting the future traffic states of a transportation network, i.e., traffic forecasting, has been the central concern of many research works in literature. The challenges mainly come from the complicated spatial and temporal correlations of traffic states. Since the transportation network naturally has a graph structure, a vast majority of existing methods are built with Graph Neural Networks Jiang & Luo (2022). In the pioneering work Diffusion Convolutional Recurrent Neural Network (DCRNN) Y. Li et al. (2018), the temporal dependencies are learned by a recurrent neural network, whose layer has an embedded module to capture the spatial correlations by a diffusion process over the road network. Later, this design paradigm has been improved from various perspectives. For example, Graph WaveNet Z. Wu et al. (2019) develops an adaptive graph to learn the spatial correlations between the sensors, and TwoResNet D. Li et al. (2022) addresses regional traffic with a low-resolution module at macroscopic scale, and the microscopic traffic is handled by a high-resolution module.

Owing to constraints in data availability, many existing methods are trained and tested with highway loop detector data, e.g., METR-LA and PEMS-Bay Y. Li et al. (2018). In urban areas, the road network has denser connections and more complicated traffic dynamics, and therefore the forecasting task is more challenging. Recently, Unmanned Aerial Vehicles (UAV) or drones are emerging as a novel solution to information collection in ITS Butilă & Boboc (2022), as they can flexibly fly over an area of interest and record high-quality videos. At the same time, the growing power of Computer Vision, Artificial Intelligence and computational resources have made it possible to automatically and efficiently extract object information from the captured videos. These advantages have empowered drones to gather rich and accurate information from a large spatial region, making them a preferable choice for urban traffic monitoring. Specifically, the pNEUMA experiments have started the understanding of urban congestion evolution through the analysis of video data captured by a swarm of drones over a large area in the city center of Athens E. Barmpounakis & Geroliminis (2020). The drone-captured videos present clear views of the monitored areas, and most vehicles on the roads can be clearly identified by computer vision-based object detection and tracking X. Wu et al. (2021). Then, traffic variables can be accurately computed for all visible roads in the field of view (FOV) E. N. Barmpounakis et al. (2019), resulting

in a great advantage compared to grounded sensors, e.g., loop detectors and CCTV cameras, whose scopes are limited to the roads with installed facilities.

The high-quality data from drones can provide solid grounds for traffic state prediction, which can be an additional asset to the existing traffic forecasting methods based on loop detector data. However, to the best of our knowledge, no public dataset has provided urban traffic data from drones and other sensors with matching areas and time spans. Besides, a new data modality will also introduce the challenge of combining multiple input sources, which can not be trivially handled by existing methods. This paper proposes to address these gaps with a novel dataset and a new baseline model for multi-source traffic prediction. In summary, the main contributions are as follows:

1. We introduce SimBarca, a novel simulation dataset that contains traffic speed data from drones and loop detectors in the city center of Barcelona. The dataset contains two prediction tasks, i.e., the prediction of regional and road segment-level speed.
2. We propose a flexible model architecture for road segment-level speed prediction, which consists of a local module for data fusion and a global module for spatial dependencies learning. We further extend the model with an additional branch for regional speed prediction.
3. We provide experiments and analysis to evaluate the model on the proposed multi-source dataset, and set an example for the evaluation of multi-source urban traffic prediction.

2 Methodology

This section will introduce the formulation of the multi-source traffic prediction problem, the process of creating the dataset from simulated vehicle trajectories, as well as the architecture of the baseline model.

Multi-Source Traffic Prediction

The traffic states of a transportation network can be measured by multiple sensors (information sources). Suppose at time t, the network traffic states $\mathbf{X}$ are observed with method m at location p, and the measured value is v. This measurement procedure can be described as

$$v = \mathcal{O}_m(\mathbf{X}, t, p), \tag{1}$$

where $\mathcal{O}_m$ generally represents the physical or statistical process to obtain the measurement.

In a real-world system, the true $\mathbf{X}$ is often unknown, and the available information from one measurement is (t, v, m, p). Let a single measurement be $s_i = (t_i, v_i, m_i, p_i)$ where i is an index. The modality indicator $m_i \in \{1, 2, ..., M\}$, and the location indicator $p_i \in \{1, 2, ..., P\}$, if there are M different modalities and P different locations in total. All data collected during a time interval is therefore a Multi-Source Time Series (MSTS) $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$.

An MSTS is considered synchronized if, for any time step with records, the measurements with all modalities on the same variable are available at all locations. Mathematically, that is $\forall\, t_j, j \in \mathbb{N}$, the equation

$$|\{(t_i, v_i, m_i, p_i) | t_i = t_j, p_i = p_k\}| = M \tag{2}$$

holds for all $p_k \in \{1, 2, ..., P\}$. However, this is not realistic mainly because different modalities can have different frequencies and spatial coverages. Besides, different locations (graph nodes) are connected by the road graph $\mathcal{G}(V, E)$, which introduces spatial correlations.

A dataset $\mathcal{D}$ is a collection of MSTS $\mathcal{S}_i$, the corresponding labels $\mathbf{y}_i$ and the road graph $\mathcal{G}$: $\mathcal{D} = \{(\mathcal{S}_1, \mathbf{y}_1), (\mathcal{S}_2, \mathbf{y}_2), ..., (\mathcal{S}_N, \mathbf{y}_N); \mathcal{G}\}$. The exact form of $\mathbf{y}_i$ will depend on the task (e.g., forecasting, interpolation and imputation), and more details will be provided in Section Experiment Settings. Suppose we use a neural network f parameterized by θ to learn to predict $\mathbf{y}$ using $\mathcal{S}$ and $\mathcal{G}$, the training loss can be generally noted as:

$$L(\theta, \mathcal{D}) = \mathbb{E}_{(\mathcal{S}, \mathbf{y}) \in \mathcal{D}} \left[l(\mathbf{y}; f_\theta(\mathcal{S}, \mathcal{G})) \right], \tag{3}$$

where l is a distance function.

Dataset Creation and Preprocessing

In this work, we generate simulation data using the microscopic traffic simulator Aimsun Casas et al. (2010) and its road network for central Barcelona. As shown in Figure 1, the network consists of many road segments and intersections. Aimsun can provide fine details of the road traffic, which allows great flexibility for subsequent processing and analysis. Concretely, there are three types of available information:

1. Network topology and the geometry of road segments and intersections.
2. Vehicle locations at each simulation time step, referenced within segments or intersections.
3. Vehicle entry and exit times for each road segment.

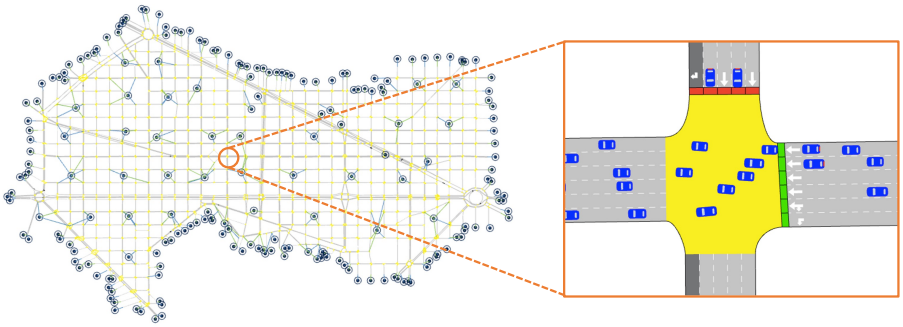

Figure 1: The road network of central Barcelona in Aimsun.

In Aimsun, the vehicles are generated according to an OD matrix, where an element means the number of vehicles going from an origin to a destination in the time period of the demand. However, the simulator provides only one such matrix, which can limit the diversity of the data. Therefore, we take the non-zero elements (leaving the zeros untouched) and apply the following random augmentations to have more diversified simulations: a) randomly set an element to zero, b) add a random percentage to the element, and c) multiply all elements by a common but random factor.

Inductive loop detectors can only measure vehicle speeds at certain points, as they are installed at fixed locations. Since vehicles can change their speeds within a road segment, the speed measurements of a loop detector often cannot represent the whole segment. On the contrary, drones can capture nearly all vehicles in their FOV, therefore it is possible to compute accurate segment-level speeds by using the vehicle trajectories. In this paper, we refer to these two types of speeds as point speed and segment speed respectively, and both of them can be extracted with the trajectory data from microscopic simulation.

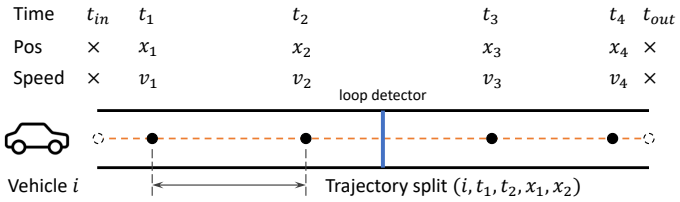

Figure 2: Available trajectory information for a road segment

Figure 2 shows an example road segment along with a vehicle trajectory. At each simulation time step ($t_1 \sim t_4$), the vehicle position and speed are known. But we try to avoid using the speed information and assume constant speed between the two known positions. Because in reality, the vehicle speeds are not directly measured from drone videos or images, but are calculated using the vehicle locations and corresponding timestamps. However, for the entry and exit points, only the timestamps are known, thus the positions have to be extrapolated from the closest known positions (e.g., the vehicle moves at speed v_1 from t_{in} to t_1).

We compose a *trajectory split* (i, t_s, t_e, x_s, x_e) using two adjacent points, which means vehicle i travels from position x_s to x_e between time t_s and t_e. Then, the trajectory splits can be grouped

by different time resolutions (e.g., every 5 s for drones and 3 min for loop detectors) to simulate the update frequency of drone data and loop detector data, or by spatial region to provide regional traffic data. For segment speed, we sum the travel distance $\Delta x = x_e - x_s$ and travel time $\Delta t = t_e - t_s$ of **all** vehicles in the road segment, and compute the segment speed as

$$\bar{v}_s = \frac{\sum \Delta x}{\sum \Delta t}. \tag{4}$$

For a loop detector installed d meters from the start. If $x_s < d < x_e$, then this vehicle is *detected*. We count the number of **detected** vehicles n in a simulation time interval, and arithmetically average their speeds to get point speed as

$$\bar{v}_p = (\sum \frac{\Delta x}{\Delta t})/n. \tag{5}$$

Using this method, we compute these two types of speeds for all road segments in the road network.

Architecture of the Baseline Model

Figure 3 shows the structure of our baseline model HiMSNet, which is a Hierarchical Multi-Source Network for traffic prediction. This architecture consists of a Global Message Exchange (GME) module to learn the spatial correlations between different locations, and a Local State Evolution (LSE) module to handle temporal dependencies and information fusion at each location.

The input data can be represented by a graph structure, where a node corresponds to a road segment, and an edge indicates the connection between two road segments. In the LSE module, the temporal encoder $\mathcal{H}_e$ will take $\mathcal{S}[:, p]$, i.e., the MSTS at location p (modalities marked in colors), and separately encode the time series of each modality into a unified representation $\mathbf{z}_p$. Then, $\mathbf{z}_p$ will go through a message encoding layer $\mathcal{R}_1$ and become a message to be shared with other locations. The GME module will then take the messages from all locations ($\mathbf{Z}$), use its encoder $\mathcal{F}_e$ and decoder $\mathcal{F}_d$ to compute the new feature with exchanged information ($\mathbf{Q}$). The message decoding layer $\mathcal{R}_2$ will take the element in $\mathbf{Q}$ corresponding to location p, and concatenate it together with the elements in $\mathbf{z}_p$ to form a joint feature $\mathbf{q}_p$. Finally, the temporal decoder $\mathcal{H}_d$ will take $\mathbf{q}_p$ and predict the future traffic states $\hat{\mathbf{y}}_p$.

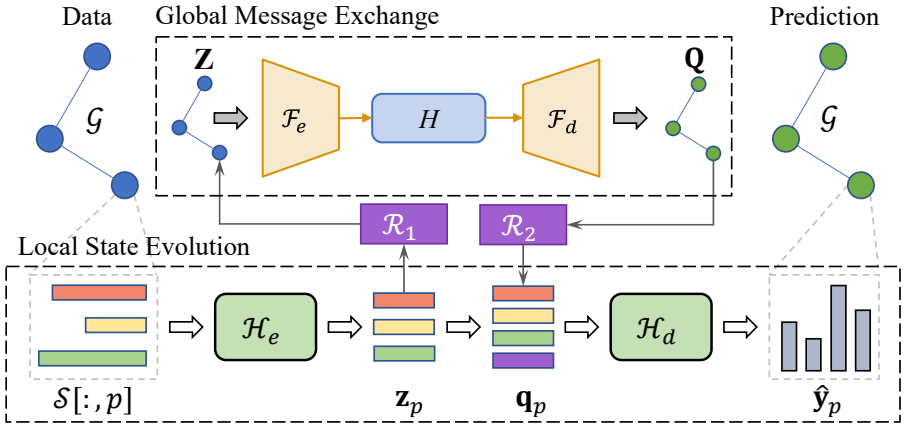

Figure 3: General structure of HiMSNet.

We extend HiMSNet with an additional branch (Figure 4) for regional traffic prediction, which can be helpful in regional traffic control. Given a defined region I, the regional branch will take the average of joint feature $\mathbf{q}_p$ for all $p \in I$ as the regional representation $\mathbf{q}_I$, and predict the regional average traffic state $\hat{\mathbf{y}}_I$ with the prediction module $\mathcal{H}_I$.

To constraint model complexity, the LSE module is shared among all locations, and similarly in the regional prediction, all regions will share the same $\mathcal{H}_I$. The training losses for both segment-level and regional prediction branches are MAE losses, and the total loss is the unweighted sum of the two branches.

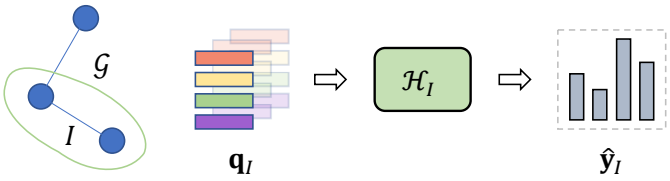

Figure 4: Regional prediction branch

3 Results and discussion

This section will provide more detailed description of the experiment settings, model implementation and evaluation.

Experiment Settings

In our experiments, each simulation session lasts for 4 hours with 0.5s time step, but only the first 90 minutes have traffic demand (including 15 minutes warm-up). That means no vehicle will be generated after 90 minutes, but the vehicles already on the road will continue to their destinations. To avoid training with nearly-empty traffic data, we truncate the simulation data 2 hours after the warm-up. Then, the samples are generated using a 1-hour sliding window, with the first half as input and the second half as prediction window. We run 101 simulations with 101 different random seeds for vehicle generation and demand matrix augmentation, and split the data into training and testing sets with a ratio of 3:1.

SimBarca provides two input modalities in a training sample: segment speed from drone observations and point speed from loop detectors. For drone speed, we sum up the vehicle travel distance Δx and travel time Δt every 5 seconds to compute the segment speed $\bar{v}_s$, using Equation 4. For loop detector speed, we set the detector at the middle points and average the speeds of all detected vehicles every 3 minutes. To construct the target sequences in segment speed prediction, we apply Equation 4 every 3 minutes in the 30-minute prediction time window, resulting in 10 prediction steps. For regional speed prediction, the same equation is applied after aggregating all road segments in the same region every 3 minutes. Both in input and output modalities, the speed is marked as -1 to indicate missing values, when no vehicle is observed during the period. We avoid using zero as the missing value because when vehicles are stopped at red lights, the calculated speed can indeed be a valid zero value.

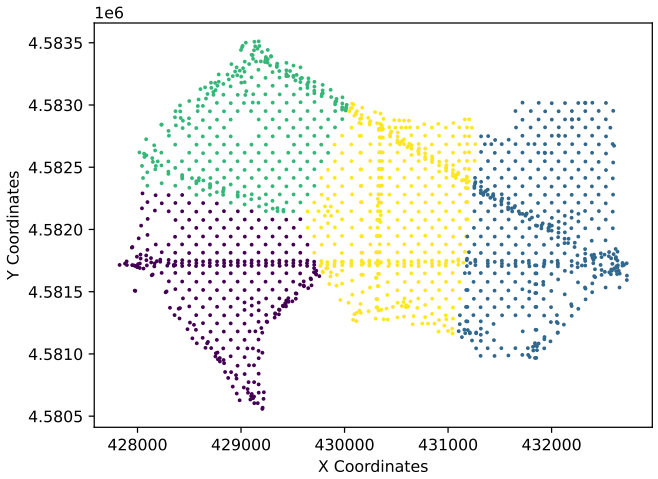

Figure 5: Clustered regions with K-Means algorithm, coordinate format EPSG:32601. A dot in the scatter plots corresponds to the center point of a road segment.

We use K-Means to cluster all the road segments into 4 regions based on spatial distances, as shown in Figure 5. In the graph-structured data, we regard a road segment as a node and a connection as

an undirectional edge. There are 1570 nodes and 2803 edges in total, and the adjacency matrix is a symmetric binary matrix. In both prediction tasks, we assume the model has "full information" and can access accurate segment speed and point speed at all locations. In future research, this assumption can be relaxed to approach real-world conditions with only partial spatial coverage.

Implementation Details

To provide a reasonable baseline model, we keep the implementation of HiMSNet as concise as possible. The loop detector and drone modalities have separate temporal encoders $\mathcal{H}_e$, they are implemented with 3-layer LSTMs with hidden size 64, whose last step output will be used as $\mathbf{z}_p$. Additionally, the $\mathcal{H}_e$ for drone modality has 2 layers of 1D convolution with kernel size 3 and stride 3 to down-sample the time resolution. Both $\mathcal{H}_d$ and $\mathcal{H}_I$ are 2-layer MLPs with hidden size 128 and output dimension 10 for the 10 prediction steps. Contrary to modality-specific temporal encoders, the decoder is shared as the features have been combined. The message encoding layer $\mathcal{R}_1$ is a 1 $\times$ 1 convolution layer that down-samples the channel dimension to 32, to alleviate computational cost at the global level. Accordingly, the message decoding layer $\mathcal{R}_2$ is also a 1 $\times$ 1 convolution layer that restores the channel dimension to 64. The GME module is implemented with 3 layers of GCNConv Kipf & Welling (2016) followed by ReLU activation. In the end, HiMSNet has 0.45M parameters, and the file size of the model is only 5.3 MB.

HiMSNet is trained for 30 epochs using Adam optimizer with learning rate 0.001, weight decay 1×10^{-4}, $\beta = (0.9, 0.999)$ and batch size 8. All implementations are based on PyTorch, and the experiments are conducted on a single Nvidia RTX 2080 Ti GPU.

Performance Evaluation

Following the common practice in traffic forecasting literature Y. Li et al. (2018), the prediction results are evaluated with three metrics: Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE). Their vector forms for a pair of prediction ($\hat{\mathbf{y}}$) and ground truth ($\mathbf{y}$) are defined as follows:

$$\begin{cases} \text{MAE} & = \frac{1}{n}\sum_i^n |\hat{\mathbf{y}}_i - \mathbf{y}_i| \\ \text{RMSE} & = \sqrt{\frac{1}{n}\sum_i^n (\hat{\mathbf{y}}_i - \mathbf{y}_i)^2} \\ \text{MAPE} & = \frac{1}{n}\sum_i^n |(\hat{\mathbf{y}}_i - \mathbf{y}_i)/\mathbf{y}_i|^2 \end{cases} . \tag{6}$$

When a road segment is blocked by upstream vehicles, its segment speed can be 0, which will cause infinite MAPE. Therefore, in this work, we only evaluate MAPE when the speed value is greater than 5 m/s, and we refer to this modified metric as MAPE*. These metrics are reported for 15 min and 30 min prediction windows, which correspond to time steps 5 and 10 in the dataset.

Figure 6 shows the progress of training loss (MAE) and testing MAE for the two tasks. Both of training losses decrease rapidly in the first a few epochs, and then gradually converge to stable values. Similar trends can be observed for testing MAE, but the convergence is slower.

(a) Segment-level (b) Regional

Figure 6: The progress of training and testing MAE for segment speed prediction

Table 1 compares the error metrics of HiMSNet with two trivial baselines, i.e., Input Average (IA) and Last Observation (LO), on both segment speed and regional speed prediction tasks. IA simply takes the per-location average over the input time window as its prediction, and this average can be based on the input source of drone or loop detector. Similarly, LO uses the last observed value for each location as its prediction. All the trivial methods have very large prediction errors, and a common pattern is that they have almost the same prediction errors for 15-min and 30-min prediction windows. This suggests the prediction task cannot be trivially handled, even with the "full information" assumption.

In the experiments for HiMSNet, "/ld" and "/drone" indicate the model is trained **without** loop detector data and drone data respectively, and the model without suffix is trained with both modalities. Compared to the trivial predictions, all the HiMSNet variants have significantly lower errors for both tasks, which demonstrates their effectiveness. The model with both modalities has the best performance for road segment prediction, and HiMSNet/ld is best for regional prediction, although they have very close error metrics. However, removing the drone modality will cause a more obvious performance drop, which suggests the drone modality is more important in these prediction tasks. Predicting the speed of individual road segments can be much more challenging than predicting the aggregated states of a region, as the former requires fine-grained information. HiMSNet has already reached 4.29% MAPE* for 30 min regional speed prediction, which is a very promising result, but the MAPE* for road segment prediction is a higher value 17.29%. Since the spatial coverage of drones can be larger than loop detectors in reality, we expect the drone modality to have a more important role in future research when the "full information" assumption is relaxed.

Table 1: Error metrics for road segment and regional speed prediction, best result in **bold**

Task	Model	15 min			30 min		
		MAE	RMSE	MAPE*	MAE	RMSE	MAPE*
Road Segment	IA_ld	5.31	6.51	43.23%	5.32	6.53	42.13%
	IA_drone	4.71	6.32	57.67%	4.77	6.37	57.99%
	LO_ld	5.59	6.91	46.50%	5.59	6.93	45.33%
	LO_drone	6.94	8.52	82.64%	6.94	8.54	82.18%
	HiMSNet/ld	**1.51**	2.54	17.40%	**1.65**	**2.75**	18.48%
	HiMSNet/drone	1.59	2.65	18.55%	1.72	2.84	19.48%
	HiMSNet	**1.51**	**2.53**	**17.29%**	**1.65**	**2.75**	**18.35%**
Regional	IA_ld	5.73	5.76	93.27%	5.85	5.95	92.41%
	IA_drone	2.55	2.72	43.04%	2.52	2.75	43.66%
	LO_ld	5.72	5.75	93.18%	5.73	5.75	93.18%
	LO_drone	2.57	2.76	43.11%	2.54	2.79	43.52%
	HiMSNet/ld	**0.27**	**0.47**	**3.27%**	**0.49**	**0.99**	4.31%
	HiMSNet/drone	0.28	0.50	3.31%	0.50	1.02	4.33%
	HiMSNet	**0.27**	0.48	**3.27%**	0.50	1.01	**4.29%**

Figure 7 illustrates the spatial distribution of the prediction MAE of HiMSNet on SimBarca test set. Most of the road segments have low prediction MAE, but there are still a few places with higher errors, which suggests the traffic states there are difficult to predict. From the perspective of traffic management, these locations are probably worth close monitoring.

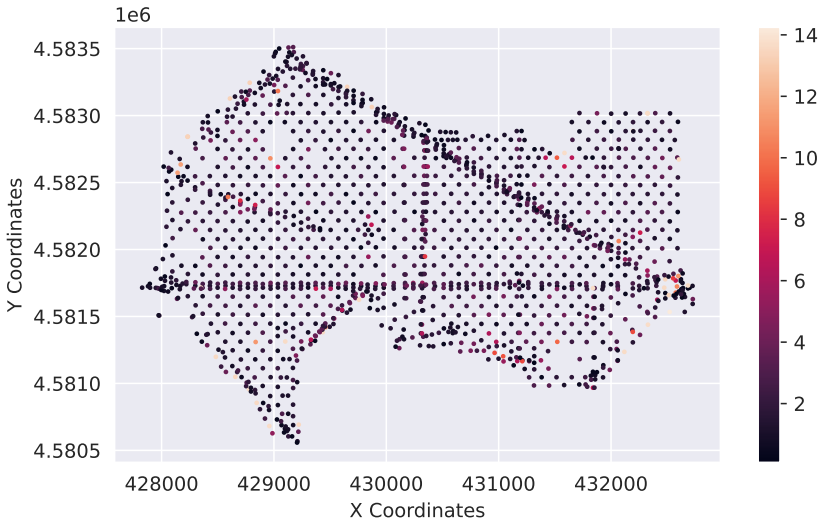

Figure 7: Average test set MAE for different locations, coordinate format EPSG:32601

4 Conclusions

In conclusion, this paper proposes a novel dataset SimBarca for urban traffic prediction, and a baseline model HiMSNet. SimBarca features multi-source input with simulated speed measurements from drones and loop detectors, and the prediction tasks include road segment-level and regional speed prediction. HiMSNet is a flexible architecture that can handle multiple input modalities and can be easily adapted to different tasks. Our experiments show that HiMSNet achieves promising results on the regional task, but the segment-level prediction is still very challenging and thus requires more exploration. We will make our codes and dataset available at `https://github.com/Weijiang-Xiong/netsanut`, and hope our work can inspire future research on multi-source urban traffic prediction.

References

Amarasingam, N., Salgadoe, A. S. A., Powell, K., Gonzalez, L. F., & Natarajan, S. (2022). A review of uav platforms, sensors, and applications for monitoring of sugarcane crops. *Remote Sensing Applications: Society and Environment*, *26*, 100712.

Barmpounakis, E., & Geroliminis, N. (2020). On the new era of urban traffic monitoring with massive drone data: The pneuma large-scale field experiment. *Transportation research part C: emerging technologies*, *111*, 50–71.

Barmpounakis, E. N., Vlahogianni, E. I., Golias, J. C., & Babinec, A. (2019). How accurate are small drones for measuring microscopic traffic parameters? *Transportation letters*, *11*(6), 332–340.

Butilă, E. V., & Boboc, R. G. (2022). Urban traffic monitoring and analysis using unmanned aerial vehicles (uavs): A systematic literature review. *Remote Sensing*, *14*(3), 620.

Casas, J., Ferrer, J. L., Garcia, D., Perarnau, J., & Torday, A. (2010). Traffic simulation with aimsun. *Fundamentals of traffic simulation*, 173–232.

Jiang, W., & Luo, J. (2022). Graph neural network for traffic forecasting: A survey. *Expert Systems with Applications*, 117921.

Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Li, D., Kwak, S., & Geroliminis, N. (2022). Tworesnet: Two-level resolution neural network for traffic forecasting of freeway networks. In *25th ieee international conference on intelligent transportation systems (itsc).*

Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2018). Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International conference on learning representations (iclr '18).*

Wu, X., Li, W., Hong, D., Tao, R., & Du, Q. (2021). Deep learning for unmanned aerial vehicle-based object detection and tracking: A survey. *IEEE Geoscience and Remote Sensing Magazine*, *10*(1), 91–124.

Wu, Z., Pan, S., Long, G., Jiang, J., & Zhang, C. (2019). Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121*.