EPJ Web of Conferences **334**, 03005 (2025) https://doi.org/10.1051/epjconf/202533403005
TGF'24

Performance of vehicle generators classifying spatially patternized vehicles in traffic in developing countries

Akihito NAGAHAMA[1,*] and *Katsuhiro* NISHINARI[2]

[1]The University of Electro-Communications, Japan
[2]The University of Tokyo, Japan

Abstract. In heterogeneous traffic observed in developing countries, certain types of vehicles tend to travel together in small clusters, known as "Groups," which can influence overall traffic flow and stability. Accurately reproducing such group behavior in traffic simulations is important for realistic modeling and for exploring ways to improve traffic flow. This study evaluated vehicle generators for microscopic traffic simulation that classify traffic entities into Groups and others (Remains) under heterogeneous and non-lane-based conditions. Three classifiers, i.e., vehicle generators, based on EDL and mEDL frameworks were compared using F_1, F_1^- (for Groups), and their product to assess the balance between detecting both categories. Classifier III achieved the most balanced performance despite being trained without explicit Group labels, indicating that it learned group-related patterns from indirect cues. While further improvements are needed for practical application, the proposed mEDL-based generator could contribute to more accurate heterogeneous-traffic simulations and provide a foundation for strategies that leverage Group behavior to enhance traffic flow.

1 Introduction

Traffic congestion remains a critical global issue, exacerbated by rapid urbanization and economic development, particularly in developing regions. The heterogeneous and often chaotic nature of traffic in these areas poses unique challenges for effective modeling and management. Addressing these challenges requires a deeper understanding of traffic dynamics and the development of robust simulation tools.

Many investigations into traffic flow heterogeneity rely on macroscopic or microscopic (e.g., agent-based) models. Macroscopic models, such as those explored by Mohan and Ramadurai [1], focus on flow characteristics by examining the proportions of different vehicle types, including motorcycles, three-wheelers, cars, and heavy vehicles. While these models simplify analysis by assuming uniform distributions of vehicle types, they often overlook critical factors such as lateral positioning biases [2]. In contrast, agent-based models capture more detailed interactions, incorporating vehicle type ratios and positional biases. However, these models frequently assume random vehicle placements, which fail to reflect real-world traffic interactions [3].

*e-mail: naga0862@uec.ac.jp

A key aspect of traffic in developing countries is the tendency of specific vehicle types to form spatial clusters. For example, Nagahama et al. [4] observed that certain types of vehicles tend to gather in heterogeneous traffic, potentially influencing flow, speed, and density depending on specific traffic conditions. Understanding these spatial arrangements, referred to as "Groups," is essential for improving the accuracy of traffic simulations and informing urban planning strategies.

Building on these findings, our previous work introduced prototype models for predicting vehicle types in heterogeneous traffic simulations, using Gaussian Process (GP) regression and Evidential Deep Learning (EDL) [5] methods. While these models showed potential in increasing prediction performance, they were unable to clearly determine whether the prototypes effectively captured the spatial patterns and randomness in heterogeneous traffic or identified other spatial features. Additionally, only four types of generators based on two model structures were developed, limiting the exploration of model parameters and structures. This restriction hindered the ability to create generators with generality, which could accurately predict vehicle types in patterned sections and generate suitable random vehicle types in other sections [6]. Furthermore, Nagahama and Nishinari [7] explored the optimization of EDL generators by varying their structures and hyperparameters. While improvements were observed in the prediction performance for motorcycles, normal passenger cars, and heavy vehicles, the models struggled with accurately replicating the spatial patterns for auto-rickshaws. This highlighted the challenge of reproducing vehicle arrangements in heterogeneous traffic and underscored the need for more detailed analyses of both patterned and random sections of traffic.

Here, we clarify the generator's tasks and its relationship with the data. The generator classifies whether a vehicle about to appear in the simulation area belongs to a Group or Remain, i.e., vehicles not belonging a Group. For a Group, it predicts the appearing vehicle type. For Remain, it predicts a random vehicle type, for example, based on proportions observed in real traffic. These predictions use the types and positions of vehicles already present in the simulation area as input. Since a Group is a collection of vehicle types that maintain a leader–follower relationship [4], the sequence of previously appeared vehicles informs the prediction. To enable these predictions, the generator requires training. Using real traffic data (vehicle types and trajectories from video data), we identify whether a vehicle belongs to a Group or Remain with the statistical method in [4]. The classification criteria for vehicles into Group or Remain are based on statistical significance, determining whether the target vehicle is part of a patterned (or frequently observed) leader–follower network sequence. These criteria consider vehicle types and spatial relationships with other vehicles. When a vehicle enters the observation area, its Group/Remain label and vehicle type serve as ground truth, while the positions and types of existing vehicles are used as inputs for training.

In particular, [6] suggested the possibility that the prototype models failed to distinguish between patternized Groups and other random vehicles, i.e., Remains. Furthermore, the authors proposed a novel EDL classifier, termed modified EDL (mEDL), in [8], which incorporates the distinction between Groups and Remains into the training data.

This study aims to address these limitations by constructing generators using both EDL and mEDL (with and without training data that includes Group and Remain distinctions), while varying their hyperparameters. The primary objective is to evaluate the classification performance, particularly in distinguishing between Groups and Remains. In this study, we evaluate which of the generators performs best instead of comparing with a certain reference model. The classification performance of Group and Remain was assessed based on ground truth data obtained from real observations as well as baseline prediction stratedies, e.g., always-positive prediction.

2 Requirements for vehicle generators

Our previous studies have revealed that certain types of vehicles, such as motorcycles, auto-rickshaws, cars, and their combinations, tend to form clusters in heterogeneous traffic [4]. These grops were referred to as Frequent Subnetwork Structures in Traffic (FSST). An FSST is a "Group" of vehicles exhibiting leader–follower relationships frequently observed over long durations in field traffic. For example, as shown in Figure 1, let us imagine that motorcycles and heavy vehicles often form pairs, where the heavy vehicle follows the motorcycle. Such pairs tend to cluster in traffic, although their occurrence may naturally increase with the number of motorcycles and heavy vehicles. FSST or Group identifies vehicle clusters that tend to gather even after accounting for the respective vehicle type numbers. Based on this observation, heterogeneous traffic can be viewed as a mix of Groups (or FSST) and Remains, as illustrated in Figure 1. Because such Groups and Remains potentially influence macroscopic traffic characteristics, such as flow, they should be replicated in microscopic simulations of heterogeneous disordered traffic.

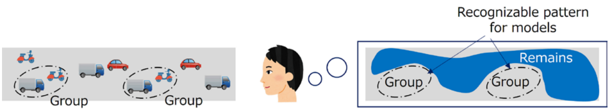

Figure 1. Viewpoint distinguishing traffic into Groups and Remains.

As shown in Figure 2, at the boundaries of microscopic heterogeneous traffic simulations, vehicle generators determine the type, timing, speed, and lateral position of the next vehicle to appear. Therefore, to replicate the observed mix of Groups and Remains, vehicle generators must possess the ability to classify whether a generated vehicle belongs to a Group or Remain based on the types and spatial arrangements of vehicles already present in the simulation (e.g., vehicles information circled in Figure 2). If the generated vehicle is classified as part of a Group, the generator must predict its type to replicate the Group accurately. Conversely, for vehicles classified as Remains, the generator must be capable of generating random vehicle types. In the case of Figure 2, the generated vehicle should be a heavy vehicle within a Group, based on the example where a pair of a motorcycle and a heavy vehicle constitutes a Group. This dual functionality enables general vehicle generation that accounts for both patterned Groups and random Remains. In other words, generator models must express uncertainty about the predicted vehicle type, indicating whether a vehicle belongs to a Group or Remain, while simultaneously outputting vehicle-type predictions.

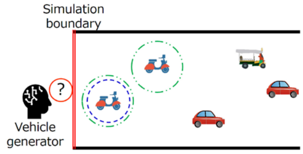

Figure 2. Schematic illustrating the input information for the generator. The positions and types of vehicles within circles are used to predict the type of vehicle entering at the entrance boundary (marked by the double line) and whether it belongs to a Group or Remains.

Machine learning models capable of providing classification results along with uncertainty include Bayesian neural networks [9], Gaussian process models [10], deep ensembles [11], etc. In addition, EDL [5] combines subjective logic with neural networks to explicitly represent uncertainty in predicted categories. EDL has demonstrated success in fields such as medical image analysis [12] and molecular discovery [13], making it a promising approach for predicting patterns and randomness in heterogeneous traffic. EDL was previously used in our proposed generator [6], but its performance in distinguishing Groups and Remains showed limitations. To address this, we proposed the use of modified EDL (mEDL) [8]. Here, Groups are vehicle structures that exhibit statistically significant differences compared to randomized leader–follower networks, making their structure and node types (vehicle types) predictable through machine learning. In contrast, Remains do not exhibit such structure and should be identified by machine learning algorithms as "unknown" or "unpredictable." EDL and mEDL are capable of distinguishing between these two types of inputs—those that allow for vehicle type prediction and those that do not. In this study, we evaluate performance of EDL and mEDL generators by exploring hyperparameters.

3 Details of EDL and mEDL generators

The generators determine the type of vehicle entering the simulation area at the entrance boundary (depicted as a double line in Figure 2). The prediction is based on the two-dimensional coordinates (x, y) and types of one to three preceding vehicles. Using this input, the generators output the probabilities $\boldsymbol{P}_\mathrm{v} = (P_\mathrm{m}, P_\mathrm{r}, P_\mathrm{c}, P_\mathrm{h})^\mathrm{T}$, where $P_\mathrm{m}, P_\mathrm{r}, P_\mathrm{c}, P_\mathrm{h}$ represent the probabilities for motorcycles (m), auto-rickshaws (r), passenger cars (c), and heavy vehicles (h), respectively. Additionally, the generators provide a measure of confidence in its predictions, i.e., uncertainty. Uncertainty serves as a measure for classifying Groups and Remains.

The EDL model, introduced by Sensoy et al. [5], incorporates subjective logic into neural networks to capture uncertainty in predictions as shown in Figure 3. Unlike standard neural networks that directly produce probabilities for each class, EDL first outputs evidence values e for the input data. These evidence values are then transformed into a Dirichlet distribution, which represents the likelihood of the input belonging to a specific class. Here, for simplicity, an example in Figure 3 is illustrated where classification is performed for two vehicle types, m and r, while simultaneously determining whether they belong to a Group or Remain.

The EDL framework applies subjective logic to calculate belief masses b for each class, including an "unknown" or Remain class u. The belief mass quantifies the model's confidence for the respective class predictions. When evidence is insufficient, the belief mass for the unknown class increases, reflecting greater uncertainty in classification. The belief masses for all classes, including the unknown class, sum to one, ensuring probabilistic consistency. Furthermore, the belief mass for any individual class, including the unknown class, is non-negative and bounded below by zero.

In EDL, if the belief mass b_u for class u exceeds the uncertainty threshold u_th, the classification result is considered uncertain, and the vehicle is classified as belonging to the Remain category. The uncertainty threshold u_th must be arbitrarily determined by the designer. Additionally, when training the EDL model, due to the architecture of the EDL, the output data in the training set cannot include the classification results of whether a vehicle belongs to a Group or Remain.

mEDL (Figure 4) [8] retains the advantages of the subjective logic framework in EDL while eliminating the need to specify the uncertainty threshold u_th. It can directly output whether the classification belongs to class u (i.e., Remain) or one of the vehicle types within the Group. Furthermore, during training, the output data in the training set can include the

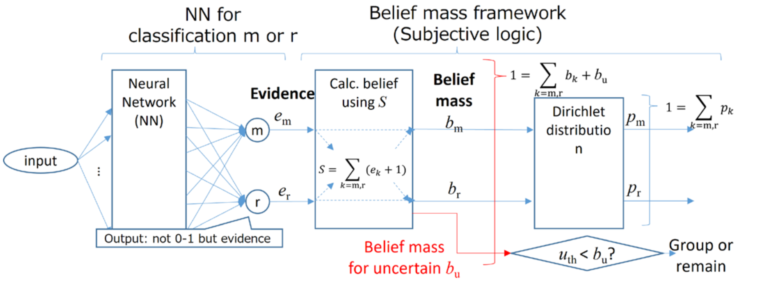

Figure 3. Structure of the EDL generator.

classification results of whether a vehicle belongs to a Group or Remain. Our previous research has been observed that this inclusion can improve the performance of class u discrimination [8].

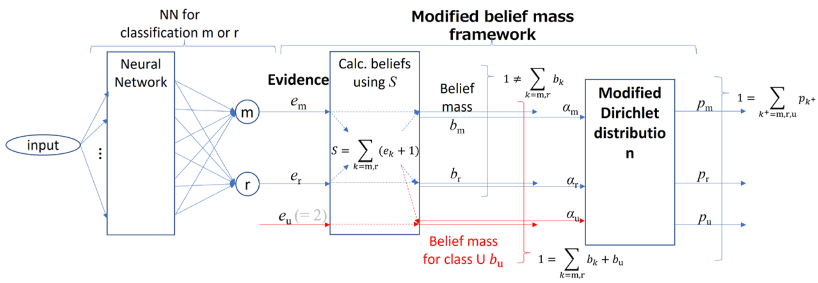

Figure 4. Structure of the mEDL generator.

A fully connected neural network was implemented using the Keras library in Python for constructing the generator models. The input data consisted of the two-dimensional coordinates of vehicles near the entrance (leading vehicles) when a new vehicle was generated into the area. The number of leading vehicles included in the input is referred to as the "leader number" hereafter. For the EDL generator, the output layer size was four, corresponding to the vehicle types: m, r, c, and h. In mEDL generators, the output layer size was extended to five to include the class u, i.e. Remain. The ReLU activation function was applied, and optimization was carried out using the Adam optimizer. Minibatch learning was conducted with a batch size of 64, an initial learning rate of 10^{-3}, and no learning rate decay.

To enhance the training process, early stopping was implemented with a maximum of 100 epochs. Model convergence was verified in all experiments, with 10% of the training data allocated for validation. The models were evaluated using a ten-fold cross-validation approach, ensuring robust performance metrics.

All experiments were executed on a system equipped with an Intel Core i7-7800X processor, an NVIDIA GeForce RTX 2080 SUPER GPU, 32 GB of RAM, and running the Windows 10 operating system.

4 Compared generators and explored hyperparameters

The performance of three types of generators was evaluated by varying the following hyperparameters. These generators included the EDL generator (Classifier I, Cl I), the mEDL generator trained with output data containing information on whether a vehicle belonged to a Group or Remain (Classifier II, Cl II), and the mEDL generator trained without such output data (Classifier III, Cl III), similar to Cl I. In other words, although the mEDL framework is capable of training on data that includes information about whether a vehicle belongs to a Group or Remain, Cl III was not provided with this information during training.

The hyperparameters varied during the evaluation were the number of hidden layers (ranging from 1 to 3), the number of nodes per layer (ranging from 4 to 64, considering only even numbers), and the leader number (ranging from 1 to 3). The above models were trained and evaluated using traffic observation data collected in Mumbai, India, in 2017 [4]. The dataset includes vehicle type information (m: motorcycles, r: auto-rickshaws, c: normal passenger cars, h: heavy vehicles) and three hours of trajectory data for each vehicle.

5 Generator performance in classifying Groups and Remains

In this study, the goal is to predict vehicle types in a simulation while distinguishing between Remains (positive examples) and Groups (negative examples). Note that there are 1095 cases of Remain and 1261 cases of Group. (Within the Group category, the distribution of vehicle types is as follows. Vehicles of type m account for 19.9%, r for 21.5%, c for 26.5%, and h for 32.1%). Therefore, instead of prioritizing one class over the other in the classification task, the ideal classifier should aim to achieve a balanced detection of both positives and negatives. Based on this perspective, the performance of Classifier I, II, and III was evaluated. The metrics considered included not only the F_1 score but also its counterpart for negative examples (F_1^-), accuracy, recall, and precision for both positive (Remain) and negative (Group) cases. Table 1 presents the top hyperparameter combinations for each classifier, selected based on the maximum value of $F_1 \times F_1^-$, along with the corresponding values of other evaluation metrics for positive and negative examples. Note that, for clarity, metrics corresponding to negative examples are indicated with a minus sign ("−"). The table also includes baseline strategies, where the classifier always predicts a single class.

Table 1. Evaluation metrics and parameter settings for each classifier and baseline strategies.

Classifier	Classifier I	Classifier II	Classifier III	Always Remain	Always Group
$F_1 \times F_1^-$	0.137	0.050	0.256	0.000	0.000
F_1	0.638	0.636	0.502	0.635	0.000
F_1^-	0.215	0.079	0.510	0.000	0.697
Accuracy	0.505	0.478	0.506	0.465	0.535
Recall	0.940	0.980	0.536	1.000	0.000
Precision	0.483	0.470	0.473	0.465	0.000
Recall−	0.127	0.042	0.481	0.000	1.000
Precision−	0.708	0.707	0.544	0.000	0.535
Leader #	3	3	1	–	–
Layer #	3	3	3	–	–
Node #	36	24	56	–	–
u_{th}	0.57	–	–	–	–

5.1 Comparison of Classifier I, II, and III

Classifier I achieves the highest $F_1 \times F_1^-$ score among its configurations with a recall of 0.940 and precision of 0.483 for Remain cases. While these values show similarity to the always-Remain baseline, its performance on Group cases improves slightly, with an F_1^- score of 0.215 and Recall− of 0.127. This configuration reflects a modest trade-off between maximizing Remain detection and introducing limited Group recognition.

Classifier II achieves its best $F_1 \times F_1^-$ score with a recall of 0.980 and precision of 0.470 for Remain cases. However, its ability to classify Groups Remains limited, with an F_1^- score of 0.079 and Recall− of 0.042. Compared to the always-Remain baseline, it shows only minor improvements.

Classifier III achieves the most balanced configuration in terms of $F_1 \times F_1^-$, with a recall of 0.536 and precision of 0.473 for Remain cases. Its Group classification performance is notably better than the other classifiers, with an F_1^- score of 0.510, Recall− of 0.481, and Precision− of 0.544. Although its overall accuracy is 0.506—only slightly above random—it demonstrates relatively even detection of both Remains and Groups.

5.2 Discussion

While all classifiers were evaluated under the same $F_1 \times F_1^-$ criterion, their behavior toward Remain and Group cases differed notably. Classifier I demonstrated strong detection of Remain cases but provided only limited improvement in Group recognition, suggesting a trade-off that still leans heavily toward the positive class. Classifier II, despite comparable scores for Remain detection, showed even weaker performance for Groups, implying that its prediction strategy may not reflect genuine understanding of Group characteristics. In contrast, Classifier III achieved the most balanced performance, with substantially improved metrics for Group classification. Although its overall accuracy was only marginally better than random, it represents a step toward fairer treatment of both classes and holds potential for practical enhancement.

One possible method to improve prediction performance is to modify the input variables. Groups were originally detected based on networks constructed using leader–follower relationships [4]. By using the leader–follower network itself as an input variable, it might become easier for the generator to distinguish between Groups and Remains. Furthermore, improving the neural network structure of the generator and its training methods remains a challenge for future work.

6 Conclusion

This study assessed the performance of vehicle generators in classifying traffic entities into Remains (positives) and Groups (negatives) within heterogeneous and disordered traffic environments in developing countries. Three classifiers, based on EDL and mEDL frameworks, were evaluated using a variety of metrics, focusing on the balance between Remain and Group detection through the product of F_1 and F_1^- scores. Among them, Classifier III achieved the highest $F_1 \times F_1^-$ score, despite being trained without explicit access to Group-related information. This suggests that the model was able to learn latent structures associated with grouping behavior from indirect cues such as time and position. In contrast, Classifier I and II showed strong performance in Remain detection but exhibited limited capability in recognizing Groups. Nonetheless, overall performance is still insufficient for practical application. Future improvements could involve directly incorporating Group-level structural information—such as leader–follower networks—into the input features, as well as refining the generator's neural architecture and training procedures. These enhancements are expected to lead to more robust models capable of both class discrimination and meaningful uncertainty estimation.

This study was supported in part by a research grant for Exploratory Research on Sustainable Humanosphere Science from the Research Institute for Sustainable Humanosphere (RISH) Kyoto University. This work was also supported by JSPS KAKENHI grant numbers 25287026, 18H05923, 19K15246, and 23K13512 and partially supported by MEXT under "Priority Issues and Exploratory Challenges on post-K (Supercomputer Fugaku)" (Exploratory Challenge 2: Construction of Models for Interaction Among Multiple Socioeconomic Phenomena, Model Development, and its Applications for Enabling Robust and Optimized Social Transportation Systems) (Project ID: hp190163).

References

[1] R. Mohan, G. Ramadurai, Transportation in Developing Economies **5**, 1 (2019)

[2] V. Kanagaraj, M. Treiber, Physica A: Statistical Mechanics and its Applications **509**, 1 (2018)

[3] D. Yang, J. Jin, B. Ran, Y. Pu, F. Yang, *Modeling and analysis of car-truck heterogeneous traffic flow based on intelligent driver car-following model*, in *Transportation Research Board 92nd Annual Meeting* (2013), 13-2358

[4] A. Nagahama, T. Wada, D. Yanagisawa, K. Nishinari, Journal of the Eastern Asia Society for Transportation Studies **14**, 1794 (2022)

[5] M. Sensoy, L. Kaplan, M. Kandemir, Advances in neural information processing systems **31** (2018)

[6] A. Nagahama, T. Wada, K. Takadama, D. Yanagisawa, K. Nishinari, K. Tanaka, *Prototype Models for Predicting Vehicle Types Generated in Heterogeneous Traffic Simulation*, in *International Conference on Traffic and Granular Flow* (Springer, 2022), pp. 431–438

[7] A. Nagahama, K. Nishinari, *Optimizing Evidential Deep Learning Generators for Modeling Vehicular Traffic Patterns in Developing Countries*, in *Proceedings of International Symposium on Scheduling 2023* (2023), OS1-1, pp. 82–86

[8] A. Nagahama, Scientific Reports **13**, 14904 (2023)

[9] D.J. MacKay, Neural computation **4**, 448 (1992)

[10] C.E. Rasmussen, C.K. Williams, Lecture notes in computer science **3176**, 63 (2004)

[11] B. Lakshminarayanan, A. Pritzel, C. Blundell, Advances in neural information processing systems **30** (2017)

[12] F.C. Ghesu, B. Georgescu, E. Gibson, S. Guendel, M.K. Kalra, R. Singh, S.R. Digumarthy, S. Grbic, D. Comaniciu, *Quantifying and leveraging classification uncertainty for chest radiograph assessment*, in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part VI 22* (Springer, 2019), pp. 676–684

[13] A.P. Soleimany, A. Amini, S. Goldman, D. Rus, S.N. Bhatia, C.W. Coley, ACS central science **7**, 1356 (2021)