Integrating Data and Rules: A Hybrid Approach for Robust Lane Change Intention Prediction under Distribution Shift

Yao Long Teng[1][0009-0000-1707-2172], Htet Naing[1][0000-0002-8014-9811], and Wentong Cai[1]

School of Computer Science and Engineering, Nanyang Technological University, Singapore

1 Introduction

This document contains supplementary materials for our paper *"Integrating Data and Rules: A Hybrid Approach for Robust Lane Change Intention Prediction under Distribution Shift"*, submitted to the 24th International Conference on Computational Science (ICCS 2024). This document contains related information mentioned in the main paper, including pseudocodes and hyperparameters of the models discussed. All the models discussed were implemented in Python.

2 Pseudocodes

As mentioned in the main paper, the Genetic Algorithm has been widely used for the calibration of various traffic simulation models [1, 4]. Without loss of generality, we use the Genetic Algorithm to calibrate our model for similar purposes. In depth pseudocode of the algorithm implemented to calibrate the hybrid model is given in Algorithm 1. The calibration of the IDM is done in a similar manner. Additionally, the physics-guided neural network for acceleration prediction has to be trained before the calibration of the hybrid model. The loss function used in the training process is given by Equation (1) as explained in the main paper. The pseudocode for this training process is given in Algorithm 2.

$$\mathcal{L}(\theta) = \frac{1}{\mathcal{N}} \sum_{i=1}^{\mathcal{N}} \left(a_i^{\text{data}} - a_i\right)^2 + \lambda \left(a_i^{\text{physics}} - a_i\right)^2 \tag{1}$$

Algorithm 1: Calibration of Hybrid model with Genetic Algorithm

```
Result: Hybrid model f_2 with parameters θ* = {P, a_thr};
Initialization:
Initialize training data Data_train;
Initialize population size N;
Initialize number of generations G;
Initialize crossover rate c_r;
Initialize mutation rate m_r;
Function InitializePopulation(N):
    population ← random set of parameters;
    return population;
Function CalculateFitness(individual, data):
    Local fitting of input-output pairs with hybrid model and its
      individual parameters;
    Compare model output to data;
    return fitness value;
Function SelectParents(population,fitness):
    Select individuals based on fitness to reproduce;
    return parents;
Function Crossover(parent1, parent2, c_r):
    Produce offspring by combining parameters of parents;
    return offspring;
Function Mutate(individual, m_r):
    Mutate parameters of individual with probability m_r;
    return mutated individual;
Procedure Main():
    population ← InitializePopulation(N);
    fitness ← CalculateFitness(population, data);
    for g ← 1 to G do
        newPopulation ← ∅;
        for i ← 1 to N do
            parents ← SelectParents(population, fitness);
            offspring ← Crossover(parents[0], parents[1], c_r);
            offspring ← Mutate(offspring, m_r);
            newPopulation ← newPopulation + offspring;
        end
        population ← newPopulation;
        fitness ← CalculateFitness(population, data);
        bestIndividual ← individual with highest fitness in population;
    end
    θ* ← parameters of bestIndividual;
return
```

Algorithm 2: Acceleration prediction model training

Result: Physics-guided NN f_1 with parameters Θ;
Initialization:
Initialize training data $Data_{train}$;
Initialize network weights Θ;
Initialize IDM calibrated with Genetic Algorithm;
Initialize desired value of $\lambda \in [0, 1]$;
Initialize batch size N;
for *epoch = 1,. . . , E* **do**
    Shuffle D_{train} to create mini-batches B_k such that $D_{train} = \cup_k B_k$;
    forall *batches* B_k **do**
        Initialize gradient accumulation: $\Delta\Theta = 0$;
        forall $(x^{(i)}, y^{(i)}) \in B_k$ **do**
            Perform forward propagation to compute prediction $\hat{y}^{(i)}$;
            Compute output $\hat{y}^{(i)}_{phy}$ with pre-calibrated IDM;
            Compute gradient of the cost function (1) w.r.t. parameters and $\nabla_\Theta J(\Theta; x^{(i)}, y^{(i)}, \hat{y}^{(i)}_{phy})$;
            Accumulate gradients: $\Delta\Theta \leftarrow \Delta\Theta + \nabla_\Theta J(\Theta; x^{(i)}, y^{(i)}, \hat{y}^{(i)}_{phy})$;
        end
        Update parameters: $\Theta \leftarrow \Theta - \alpha \cdot \frac{\Delta\Theta}{|B_k|}$
    end
end

3 Related parameters and bounds

In this section, we present the parameters and bounds used in our experiments. Bounds used for calibration of IDM and MOBIL were derived from existing studies [3, 2, 6, 5, 7] and presented in Table 1 and Table 2 respectively.

Table 1. IDM parameter bounds

Parameter	Bound
Maximum desired velocity, v_0	[20,80] m/s
Desired time headway, T	[0,5] s
Jam spacing distance, s_0	[1,5] m
Maximum acceleration, acc_{max}	[0,5] m/s^2
Maximum deceleration, $dacc_{max}$	[0,5] m/s^2

Additionally, the parameters used in the neural network trained for acceleration prediction are given in Table 4, while the parameters used in the calibration with Genetic Algorithm are as shown in Table 3.

Table 2. MOBIL parameter bounds

Parameter	Bound
Politeness factor, p	[0,1]
Acceleration threshold, a_{thr}	[-4,4] m/s^2

Table 3. Genetic Algorithm Parameters

Parameter	Value
Population Size	60
Cross Over Rate	0.5
Mutation Probability	0.3
Mutation Noise	0.10
Mutation Method	Uniform noise addition
Proportion of Parents in Population	0.5
Maximum number of generations	100
Crossover Method	Uniform
Crossover Point	1
Selection Method	2-way tournament

Table 4. Neural network parameters

Parameter	Value
Batch Size	32
Learning Rate	0.001
Initialization type	Xavier
λ used in loss function	0.5

References

1. Cunha, A.L., Bessa, J.E., Setti, J.R.: Genetic algorithm for the calibration of vehicle performance models of microscopic traffic simulators. In: Progress in Artificial Intelligence: 14th Portuguese Conference on Artificial Intelligence, EPIA 2009, Aveiro, Portugal, October 12-15, 2009. Proceedings 14. pp. 3–14. Springer (2009)
2. Kesting, A., Treiber, M., Helbing, D.: General lane-changing model mobil for car-following models. Transportation Research Record **1999**(1), 86–94 (2007)
3. Khelfa, B., Ba, I., Tordeux, A.: Predicting highway lane-changing maneuvers: A benchmark analysis of machine and ensemble learning algorithms. Physica A: Statistical Mechanics and its Applications p. 128471 (2023)
4. Kim, K.O., Rilett, L.R.: Genetic-algorithm based approach for calibrating microscopic simulation models. In: ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 01TH8585). pp. 698–704. IEEE (2001)
5. Mo, Z., Shi, R., Di, X.: A physics-informed deep learning paradigm for car-following models. Transportation research part C: emerging technologies **130**, 103240 (2021)
6. Naing, H., Cai, W., Wu, T., Yu, L.: Dynamic car-following model calibration with deep reinforcement learning. In: 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC). pp. 959–966. IEEE (2022)
7. Treiber, M., Hennecke, A., Helbing, D.: Congested traffic states in empirical observations and microscopic simulations. Physical review E **62**(2), 1805 (2000)