Towards Comprehensive Maneuver Decisions for Lane Change Using Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this paper, we consider the problem of autonomous lane changing for self driving vehicles in a multi-lane, multi-agent setting. We use reinforcement learning solely to obtain a high-level policy for decision making, while the lower level action is executed by a pre-defined controller. To obtain a comprehensive model adaptive to as wide traffic scenarios as possible, training is carried out based on more than 700 handcrafted traffic scenarios with various types of traffic involved. A new asynchronous DQN architecture is proposed to handle the training samples' diversity while improving the training efficiency. Moreover, we also present a new state representation that contains both short range information and long range information, to retain the merit of each individual representation while refining them in terms of generalization ability and training efficiency. The generated policy is evaluated on other 200 testing scenarios in a simulator, the results demonstrate that our approach shows the better generalization ability than a rule-based baseline, and possesses better intelligence and flexibility.

1 Introduction and Related Work

In recent years there has been a growing interest in self driving cars due to the high potential in leading to more efficient road networks and more convenient user experience. Lane change making is one of the fundamental skills that a self driving car must possess, through reasoning of interactions with other agents and forming an efficient long term strategy adaptive to various traffic, which makes this problem much challenging.

The approaches for maneuver decisions cab be classified into three categories. Early approaches focused on handcrafted rules typically realized in large state machines, each requiring thoughtful engineering and expert knowledge [1][2][3]. Recent work focuses on more complex ways with additional domain knowledge to predict and generate maneuver decisions [4][5][6]. Recently, as the development of deep reinforcement learning (DRL), there have been a few studies highlighting lange change decision making through DRL for highway scenarios in simulation[7][8][9].

Great progress has been made in the field of using DRL based approach as discussed above, nevertheless there are still several challenging issues needed to be solve in certain aspects. First and most important, existing works only focus on training and testing the lane change policy on the same type of traffic (e.g., similar distribution of social vehicles), yet neglecting the transfer ability to different traffic scenarios, which appears virtually impossible to be put into use practically. Although it is commonly recognized that generalization ability plays an important role, especially for the case of autonomous driving, it still remains challenging. Next, current studies employ either global grid or relational grid as the state representation, both of which give rise to some limitations. The former method posses challenged generalization ability to scenarios with different lane number and suffers from slow convergence due to large volume of network parameters, while the latter solution only takes short range view into account without distinguishing different traffic status, which might lead to the myopic policies and be difficult to adapt to various traffic scenarios.

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

Thus, in this paper we re-investigate the strengths of deep reinforcement learning for high-level decision making in multi-lane and multi-agent settings, trying to address the issues mentioned above with the main contributions summarized as follows: firstly, to obtain a comprehensive model adaptive to as wide scenarios as possible, training is carried out based on more than 700 handcrafted traffic scenarios with various types of traffic involved. Meanwhile, a new asynchronous DQN architecture is proposed to better exploit the sample diversity and improve the training efficiency. Secondly, we propose a new state abstraction that takes both short range information and long range information into consideration, in combination with a flexible reward design. The primary goal is to retain the merit of each individual solution while refine them in terms of generalization ability, training efficiency as well as adaptive capacity. We present preliminary benchmarks and demonstrate that our solution can significantly outperform a rule-based greedy baseline from the generalization ability. And it is particularly worth mentioning that we have experimented our model in reality under urban highway scenarios in Shanghai, China. We remark that, to the best of our knowledge it is the first attempt for DRL-based lane change solution to be tested in real traffic.

The rest of this paper is organized as follows: the reinforcement learning solution with novelty is demonstrated in Section 2, with detailed experiment setup and implementations presented in Section 3. Our approach is evaluated in comparison with a baseline solutions in Section 4. Conclusion and Future Works are summarized in Section 5.

2 Reinforcement Learning Solution

Intuitively, learning a high-level decision making policy can be modeled as a reinforcement learning problem, which attempts to maximize the cumulative rewards through continuously interaction with the environment. In most existing works, the learned models are trained and tested only under a fixed type of traffic with the relatively good performance. However, for a policy to be put into use practically, it should perform equivalently well in various traffic types, which raise a huge challenge for training: how to train a policy with both higher generalization ability and lower time cost? We attempt to handle these problems from the aspect of both the state representation and the algorithm architecture.

2.1 State Representation

For a multi-task problem, due to the large volume of data coming from different tasks, the state space should be designed in consideration of the representation of the real environment, the training efficiency, as well as the transfer ability to new tasks. To be specific, (1) state representation should be abstract enough and independent of the factors such as lanes' number, the road geometry and the number of surrounding cars. (2) Low dimension of representation is recommended aiming at fast convergence. (3) Traffic status should also be considered as indicators of current environments to distinguish different scenarios. Based on above considerations, we pay attention to three classes of information: the ego information, the short range information, and the long range information, denoted as S_e, S_{short} and S_{long} respectively. Ego information only includes the speed v of the ego car. Short range information is a small number of key perception indicators that directly represent the relative relationship between the ego car and its surrounding cars [10]. It consists of the social cars' states inside the local scope of the ego car, which is independent of the road geometry, the number of lanes, or the number of social cars involved in the traffic. It is intuitive that surrounding cars play an crucial role in lane change making and it has also been reported in [9] and [10] that the short range information works well for the fixed type of traffic.

Despite of the major role of short range information, there still exists problems for the induced policy to be generalized from scenarios to scenarios since it does not distinguish different types of traffic. For instance, lane change behavior in dense traffic is more risky than that in sparse traffic, but such difference can not be handled only given the short range information. As a consequence, we add some high level description of traffic status, named as long range information, aiming at treating various types of traffic discriminatively. Long range information usually involves all the information of cars which can be perceived on the road, which may contribute to learn a more intelligent and foresighted strategy.

2.2 Algorithm Architecture

To deal with the efficiency of large-scale training with various scenarios, a new asynchronous DQN architecture is proposed. The asynchronous DQN involves one master node and many slave nodes with the algorithms demonstrated in Algorithm 1 and Algorithm 1, a shared queue is used to achieve communication. The master node is used for training an unified model, while the slave nodes just focus on collecting the date using the model during the interaction with environment. This kind of architecture is designed by the following two motivations: 1) The most time-consuming part of the online training process is the interaction of agent with the complex environment to sample data, which is typically representative for autonomous driving task. Our proposed asynchronous DQN approach can significantly speed up the process of data generation. 2) While the traditional DQN is trained on the data flow from one kind of traffic type to another, the proposed asynchronous DQN can be fed with the data from various traffic types at one time, which can enhance the generalization ability of the desired model.

Although existing asynchronous framework, like A3C [11], has the similar consideration, our architecture enjoys advantages in the following two aspects: first, the parallelization of data generation is implemented in multi-process level instead of multi-thread level, which makes the architecture much more scalable to multiple machines. Consequently, there also exist great potentials for improving the exploration efficiency. Second, the model is only updated in the master node without conducting the synchronization between local and remote, thus the memory overhead and communication cost will be reduced dramatically.

Algorithm 1 Asynchronous DQN - pseudocode

Pseudocode for each slave node.

Initialize process update counter $t = 0$
Get initial state s
repeat
  repeat
    Choose a scenario randomly from the training set as the environment for current episode
    With probability ε select a random action a
    if Model output is selected **then**
      Send current state s to the master node
      Receive $Q(s; a; \theta)$ from the master node and execute $\max_{a'} Q(s; a'; \theta)$
    end if
    Receive new state s' and reward r
    Send the transition (s, a, r, s') to the master node
    $s = s'$
    $t \leftarrow t + 1$
  until s is terminal
until $t > t_{max}$

Pseudocode for the master node.

Initialize replay buffer $\mathcal{B}$ with capacity N
Initialize the parameters θ of action value network $Q(s; a; \theta)$
Initialize target network $Q(s; a; \theta^-)$ with $\theta^- \leftarrow \theta$
Initialize training update counter $t = 0$
repeat
  Fetch the data D from the slave node
  if Data type is not TRAIN **then**
    Get s from data D to compute $Q(s; a; \theta)$ and send $Q(s; a; \theta)$ to the corresponding slave node
  else
    Get the transition (s, a, r, s') from data D and put it into $\mathcal{B}$
    Sample random minibatch of transitions (s, a, r, s') from $\mathcal{B}$
    For each transition in the minibatch, set

$$y = \begin{cases} r, & \text{for terminal } s', \\ r + \gamma \max_{a'} Q(s', a', \theta^-), & \text{for non-terminal } s', , \end{cases}$$

    Perform a gradient descent on $\Sigma_{minibatch}(y - Q(s, a, \theta))^2$
    if t mod $I_{target} == 0$ **then**
      Update the target network $\theta^- \leftarrow \theta$
    end if
    $t \leftarrow t + 1$
  end if
until $t > t_{max}$

3 Experimental Setup

3.1 Scenario Configuration

In this section, we formally define our experiments' environment. Our goal is to generate a comprehensive model which is adaptive to various kinds of traffic. For this, a multi-lane multi-agent setting with the urban traffic environment setup is considered in our self-developed simulator KyberSim, which is similar to the commonly used traffic simulator SUMO and equipped with realistic kinematics and dynamic for the ego car. To be specific, more than 700 scenarios as training scenarios are constructed, and each scenario is a $4km$-long track consisting of 3 lanes where the social cars are driving. The numbers of social cars involved in the traffic is randomly chosen in $[0, 100] \cap \mathbf{Z}$. The average traffic speed of one scenario is set to $V_e = \{5km/h, 10km/h, 20km/h, 30km/h, 40km/h, 50km/h, 60km/h, 70km/h\}$, with the variance set to $V_\sigma = \{3km/h, 5km/h, 8km/h, 10km/h, 12km/h\}$. Denote "gap" as the distance between two adjacent social cars in the same lane, then the initial average gap of one scenario is set to $gap_e = \{30m, 50m, 80m, 100m, 120m, 150m, 200m\}$, with the variance set to $gap_\sigma = \{10m, 30m, 50m, 60m, 80m\}$. For the sake of simplicity we do not allow any social car to change lanes. The distance between the ego car's starting location and the track's starting point is distributed randomly in $[0m, 200m]$ and the ego car locates randomly in the three lanes at the initial time. From such design, various kinds of traffic scenarios emerge including "super slow dense ","slow dense","intermediate dense", "fast dense", "slow sparse", "intermediate spare", "fast spare", "slow uniform ", "intermediate uniform","fast uniform", and so on. We also construct 200 other scenarios as testing scenarios, each of which belongs to one of the existing traffic types listed above.

We also incorporate an optimization method and a controller in the low-level module to generate a sequence of low-level actions that can make a car change between adjacent lanes or follow the front car while avoiding collisions. Since collisions are never allowed during training or testing, the learning process truly focuses on learning only the efficiency and generality of the high level strategy.

The performance of the lane change making policy will be evaluated on the testing scenarios in terms of the average speed and the average number of lane change for passing the track under different types of traffic. The model with higher average speed and lower number of change is better.

3.2 Implementations

Outputs For lane change maneuvers, we break down the high level decisions into the following 3 actions that can be taken at any time step: (1) **0**: go straight, (2) **1**: turn left, (3) **2**: turn right. The ego car will execute **0** for fixed 20 simulator steps and **1** or **2** for 60 to 120 simulator steps, according to the command timing.

Inputs The specific inputs in our implementation are shown in Table 1, where gap is the longitudinal distance between two adjacent cars in the same lane. It is intuitive that all gaps and speeds involved on road can jointly reveal the key characteristics of the traffic, which are therefore feasible as long range information. The non-perceivable lane or missing entities are indicated by a dedicated value. To obtain a more flexible perception scope, relative distance and speed to the ego car is adopted, whose ranges are set to be $[0, 12.5]$ and $[0, 2]$ respectively.

Rewards Design The overall design philosophy of the reward function is to encourage the higher speed with the least number of lane change, since changing itself is risky and time-consuming compared with following behavior in reality. In addition, the degree of punishment should vary adaptively as the environment changes to conform to the actual situations.

For a transition (S, a, S'), the reward function is defined as

$$R(S, a, S') = \begin{cases} V', & \text{if } a = 0, \\ c \cdot \kappa_l \cdot \kappa_g \cdot V', & \text{if } a \neq 0, \end{cases} \tag{1}$$

where $c = 0.3$ is the penalty factor for change action and κ_l and κ_g are shrinkage coefficients concerning the short-term gain and the long-term benefit for change respectively. κ_l is computed as T_e/T, where T is the time cost for executing action a and T_e is the average changing cost in the recent time window, where the window size is set to 500. We remark that in KyberSim and of course in real environments, the ego car will spend distinctly more time changing independently of its speed

Table 1: Inputs of the Network

Ego information	
v	Speed of the ego car
Short range information	
p_dist, p_v_L	longitudinal distance and speed of the preceding car in the left lane
p_dist_M, p_v_M	longitudinal distance and speed of the preceding car in the current lane
p_dist_R, p_v_R	longitudinal distance and speed of the preceding car in the right lane
f_dist_L, f_v_L	longitudinal distance and speed of the following car in the left lane
f_dist_M, f_v_M	longitudinal distance and speed of the following car in the current lane
f_dist_R, f_v_R	longitudinal distance and speed of the following car in the right lane
Long range information	
gap_L	average gap of all cars in the left lanes
gap_M	average gap of all cars in the current lane
gap_R	average gap of all cars in the right lanes
v_L	average speed of all cars in the left lanes
v_M	average speed of all cars in the current lane
v_R	average speed of all cars in the right lanes

if the timing is locally unreasonable, since the ego car has to slow down to follow the car in the goal lane. It can been seen that κ_l encourages locally reasonable change while penalize the opposite case, which plays a role from a short range perspective. κ_g is computed as gap_G/gap_M, where gap_G and gap_M are defined in Table 1. It is a way of measuring the sparsity gain, aiming at encouraging the change from the denser lane to the smoother side and penalize the opposite case, which is designed from a long range perspective. Using the product of κ_g and κ_l, mutual compensation is produced to obtain a synthetically optimal policy.

Network We use a network with solely fully-connected layers, including two hidden layers with 300 and 600 neurons respectively. On the output layer there is a neuron for each action. The given value for each action is its estimated Q-value.

Hyper Parameters The network is trained with 1 million iterations. The number of actor processes is set to 6. The discount factor $\gamma = 0.3$, the batch size for one update is 96, the update period for the target network is $I_{target} = 1000$ and the capacity of $N = 300000$. We adopt ε-greedy exploration, where ε is annealed from 1.0 to 0.1. As optimization method for our DQN we use the Adam algorithm with a learning rate of 10^{-5}.

4 Evaluation

We use four different traffic types for assessing the properties of the proposed approach. 200 testing scenarios are grouped into 4 traffic types including “super dense", “dense",“uniform" and “sparse". Each type consists of a dozen of scenarios with different average speeds for social cars. For each traffic type, we test our approach under all its member scenarios for 5 times to obtain the average score.

Table 2: Comparison between our approach and a greedy rule-based baseline on 4 traffic types.

	Super Dense		Dense		Uniform		Sparse	
	Baseline	Ours	Baseline	Ours	Baseline	Ours	Baseline	Ours
Avg Speed(km/h)	6.46	6.44	11.8	12.02	51	49.6	58	67
Ave Count	473	14.4	61.8	12	12	18.6	4.8	6.2
Soft Changes Per(%)	0	15	0	0	0	9	0	30

We first compare our approach with a rule-based greedy baseline, where the ego car makes a lane change once there exists a social car in front of the ego car for more than 2 seconds, and at the same time, the front car in the neighbor lane is further to the ego car.

The benchmark results are summarized in Table 2. In non-dense traffic, our approach is able to achieve almost the same or higher average speed compared with the baseline with relatively more changing activities, and in dense traffic, the advantage is more obvious that the baseline policy makes too many unnecessary lane change decisions. From this we can see that rule-based polices are hard to generalize to different types of traffic even though it can be designed perfectly for some fixed traffic type. To evaluate the intelligence of our model, we also record the percentage of “soft unreasonable" changes, which means, the ego car is in the right most or left most lane and the preceding car in the goal lane is nearer than the preceding car in the current lane. It can been seen that such changes might

be locally unreasonable yet potentially globally reasonable. From Table 2, our approach enjoys some percentage of soft changes while the baseline does not, demonstrating the long term planning ability of our approach. Figure 1 is an example to illustrate the superiority of the “soft changes", that the ego car sacrifices a short term gain to optimize its long term return.

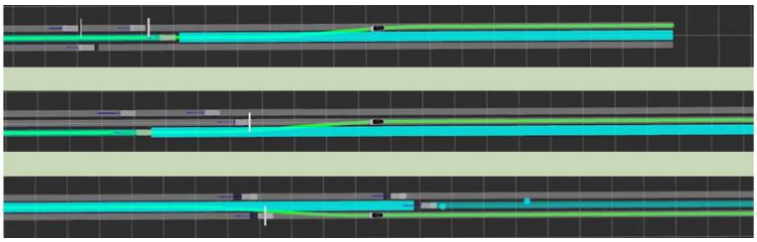

Figure 1: (upper) The ego car is driving in the right most lane and the action change left is made, which is locally unreasonable; (middle) Subsequently, the ego car makes left change again; (lower) After following for a while in the left lane, the ego car changes back to the middle lane. By executing above three processes together a higher speed can be achieved.

Long Range Information Effects To evaluate the performance gain brought in by global information, we add a **pure short approach** and a **pure long approach** for comparison, where the input of the former is $S = (v, S_{short})$ and they share the same reward function with $\kappa_l = \kappa_g = 1$.

Total training set is used for training and the implementations are the same as introduced in Section 3.2. It can be seen from Figure 2 that the long range solution enjoys much better performance than short range one, even though they share the same reward function. In addition, flexible reward design can both improve the performance of the model and speed up the training process significantly.

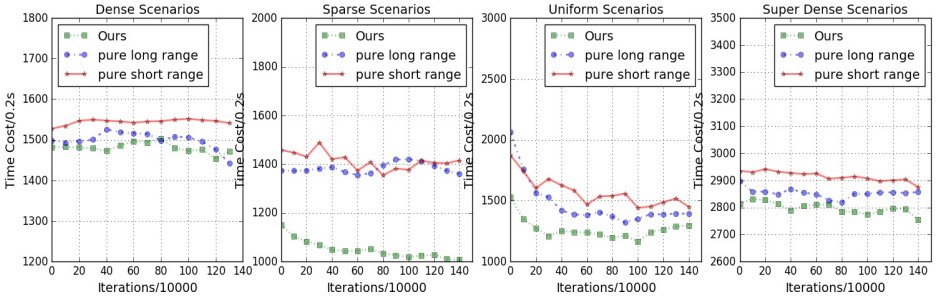

Figure 2: Learning speed comparison for 3 approaches

5 Conclusion and Future Works

In this paper, we investigate the problem of autonomous lane changing for self driving vehicles in a multi-lane, multi-agent setting. Reinforcement learning method is adopted for training a high-level policy for tactical decision making. To obtain a comprehensive model adaptive to as wide traffic types as possible, training is carried out based on more than 700 handcrafted traffic scenarios with various types of traffic involved. A new asynchronous DQN architecture is proposed to handle the training sample diversity while improving the training efficiency. Moreover, we also propose a new state representation that contains both short range information and long range information, aiming at retaining the merit of each individual representation while refining them in terms of generalization ability, training efficiency as well as adaptive capacity. The generated policy is evaluated on 200 other testing scenarios in a simulator, the results demonstrate that our approach enjoys the better generality ability then a rule-based baseline, and posses better intelligence and flexibility. Of course, there still exist some improvements in future, for instance, the comparison between our asynchronous DQN architecture and A3C framework needs to be conducted, the benefits brought in by long range information need to be tested on other self-driving applications.

References

[1] C. Urmson, J. Anhalt, D. Bagnell, et al., “Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[2] A. Bacha, C. Bauman, R. Faruque, et al., “Odin: Team victortango’s entry in the darpa urban challenge," *Journal of field Robotics*, vol. 25, no. 8, pp. 467–492, 2008.

[3] J. Leonard, J. How, S. Teller, et al., “A perception-driven autonomous urban vehicle," *Journal of Field Robotics*, vol. 25, no. 10, pp. 727–774, 2008.

[4] S. Ulbrich and M. Maurer, “Towards tactical lane change behavior planning for automated vehicles," in *18th IEEE International Conference on Intelligent Transportation Systems, ITSC*, 2015, pp. 989–995.

[5] A. Lawitzky, D. Althoff, C. F. Passenberg, et al., “Interactive scene prediction for automotive applications," in *Intelligent Vehicles Symposium (IV)*, IEEE, 2013, pp. 1028–1033.

[6] M. Bahram, A. Lawitzky, J. Friedrichs, et al., “A game-theoretic approach to replanning-aware interactive scene prediction and planning," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 3981– 3992, 2016.

[7] Mirchevska B, Blum M, Louis L, et al., “ Reinforcement learning for autonomous maneuvering in highway scenarios",[C] *Workshop for Driving Assistance Systems and Autonomous Driving*. 2017: 32-41.

[8] M. Mukadam, A. Cosgun, A. Nakhaei, and K. Fujimura, “Tactical decision making for lane changing with deep reinforcement learning" , 2017.

[9] Peter Wolf, Karl Kurzer, Tobias Wingert, Florian Kuhnt and J. Marius Zollner,“ Adaptive Behavior Generation for Autonomous Driving using Deep Reinforcement Learning with Compact Semantic States," *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 993-1000, 26-30 June 2018.

[10] Chen C, Seff A, Kornhauser A, et al. “Deepdriving: Learning affordance for direct perception in autonomous driving,"[C] *Proceedings of the IEEE International Conference on Computer Vision.* 2015: 2722-2730.

[11] Mnih V, Badia A P, Mirza M, et al. “ Asynchronous methods for deep reinforcement learning,"[C] *International conference on machine learning.* 2016: 1928-1937.