Neuro-Symbolic Program Search for Autonomous Driving Decision Module Design

Jiankai Sun[1] **Hao Sun**[1] **Tian Han**[2] **Bolei Zhou**[1]
[1]The Chinese University of Hong Kong [2]Stevens Institute of Technology
{sj019, sh018, bzhou}@ie.cuhk.edu.hk than6@stevens.edu

Abstract: As a promising topic in cognitive robotics, neuro-symbolic modeling integrates symbolic reasoning and neural representation altogether. However, previous neuro-symbolic models usually wire their structures and the connections manually, making the underlying parameters sub-optimal. In this work, we propose the Neuro-Symbolic Program Search (NSPS) to improve the autonomous driving system design. NSPS is a novel automated search method that synthesizes the Neuro-Symbolic Programs. It can produce robust and expressive Neuro-Symbolic Programs and automatically tune the hyper-parameters. We validate NSPS in the CARLA driving simulation environment. The resulting Neuro-Symbolic Decision Programs successfully handle multiple traffic scenarios. Compared with previous neural-network-based driving and rule-based methods, our neuro-symbolic driving pipeline achieves more stable and safer behaviors in complex driving scenarios while maintaining an interpretable symbolic decision-making process.

Keywords: Neuro-Symbolic AI, Cognitive Robotics, Autonomous Driving

1 Introduction

The standard pipeline of the autonomous driving system includes four components: perception, decision, planning, and control [1]. With the significant progress of deep learning models, recent works [2] propose to learn to drive by training deep neural network in an end-to-end manner. Deep neural network works well on a certain type of reactive tasks in autonomous driving, such as object detection and land segmentation. However, they can fail as a result of noisy sensor signals [3] or physical adversarial samples [4]. Thus it remains challenging to integrate the deep learning methods in high-stakes driving systems because of the stability and robustness concerns.

On the other hand, the current deep neural networks often lack the capability for abstract reasoning, which is crucial for decision-making in autonomous driving. On the contrary, symbolic representations are ideal for modeling high-level reasoning and decision making in autonomous driving. Furthermore, autonomous driving is a task subject to strong rules such as traffic regulation and speed limit, where the symbolic systems with logic preconditions can easily incorporate such prior knowledge and constraints. For this motivation, we aim at exploring the neuro-symbolic autonomous driving system, which is able to unify the generalizable connectionist learning and the interpretable symbolic reasoning together. One challenge for implementing the neuro-symbolic driving program is the manual design process, which is time-consuming and prone to constructing sub-optimal programs. Therefore, it is crucial to improve the level of automation in designing neuro-symbolic programs. *Neuro-Symbolic Program Search (NSPS)* is defined as a task to automatically search from the given neuro-symbolic operation sets, select the necessary neuro-symbols, and assemble them into a program termed as *Neuro-Symbolic Program (NSP)*. NSP is end-to-end differentiable and highly generalizable, and can be optimized to achieve better performance compared to manually wired programs. Thus, we replace the traditional decision module in the autonomous driving pipeline with *Neuro-Symbolic Decision Program (NSDP)* searched by NSPS. Furthermore, because of the neuro-symbolic representation, the whole decision process becomes much more transparent and understandable, compared to the black-box property of deep neural networks.

4th Conference on Robot Learning (CoRL 2020), Cambridge MA, USA.

```
# Numerical Stream:
Vel, Accel, BBox, Waypoint = Filter(input)
# Logical Stream:
(1) ((MinVelocity > 0) ∧ (DistanceIntersection < d0) →
      ApproachingIntersection
meaning:  if the car is moving and heading towards an
      intersection since it has been deemed close to
      the intersection, then the car is approaching the
      intersection.
(2) ApproachingIntersection ∧ (CurrentLane = Right) →
      DecelerationPhase:
meaning:  the car is approaching an intersection and
      currently in the right lane.
# Query
(3) target_velocity, target_waypoint_index = Query(Vel,
      Accel, BBox, Waypoint, DecelerationPhase)
meaning:  query target velocity and waypoint index
      according to given information and current phase.
```

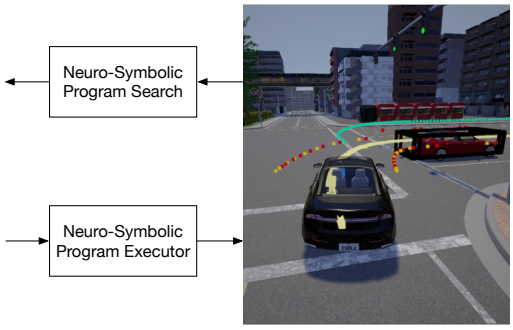


Figure 1: **A turning-right scenario for Neuro-Symbolic Driver.** On the right is the third-person view of Neuro-Symbolic Driver, and on the left is the pseudo-code of an example Neuro-Symbolic Decision Program (NSDP), showing that the Deceleration Phase is selected in the current scene. NSDP queries the waypoint index and target velocity in the deceleration phase, which further implements the order of performing a right turn.

In order to describe the driving decision with symbolic representations, we define a domain-specific language (DSL) for autonomous driving, containing both basic primitives for parts with driving attributes (*e.g.*, vehicle velocity, acceleration, pose, etc.), as well as statements such as if-else to enforce higher-level priors. A framework is proposed for searching Neuro-Symbolic Decision Program (NSDP) to effectively integrate learning and reasoning. For example, given the turning-right driving scenario in Figure 1, humans are able to instantly decide at which stage the vehicle is. Beyond these intuitive decisions, NSDP needs to give specific instructions (*e.g.*, target waypoint index, target velocity) to the downstream motion planner and controller. Since defining a fixed and known optimizable reward function that inculcates the desired behavior can be challenging for autonomous driving [5], we apply the obtained NSDP to GAIL [6] and learn from demonstration (LfD) for autonomous driving.

We summarize our contributions as follows:

- A novel program search framework *Neuro-Symbolic Program Search (NSPS)* is proposed to improve the autonomous driving system design, which can synthesize end-to-end differentiable *Neuro-Symbolic Programs*, by combining neuro-symbolic reasoning with representation learning.
- A domain-specific language is designed for differentiable neuro-symbolic behavior, to specify all the behaviors for reactive and deliberative autonomous driving.
- Experiments show the *Neuro-Symbolic Decision Program* resulting from our method is able to be generalized to various driving scenarios, allowing domain knowledge such as traffic rules to be encoded in the model to handle uncertainty.

2 Related Work

Neuro-Symbolic Program Synthesis Neuro-symbolic systems provide a unified foundation for learning and efficient reasoning and can meet the need for robustness, which are crucial to autonomous driving applications. Neuro-symbolic system has been successfully applied in the realm of visual question answering [7, 8] and cognitive robotics (e.g. autonomous driving [9], navigation [10]). Symbolic Reinforcement Learning [11] offers a better balance between generalization and specialization for decision making. Neuro-Symbolic Program Synthesis [12] typically performs some form of search over the space of programs to generate a program that is consistent with a variety of constraints [13]. We formulate the Neuro-Symbolic Program Search as a stochastic optimization problem, which can not only efficiently search the program architecture but also search corresponding program hyperparameters that are previously manually crafted.

Autonomous Driving System A widely adopted hierarchical structure design of an autonomous driving pipeline could be traced back to DARPA Urban Challenge [1, 14], which is composed of

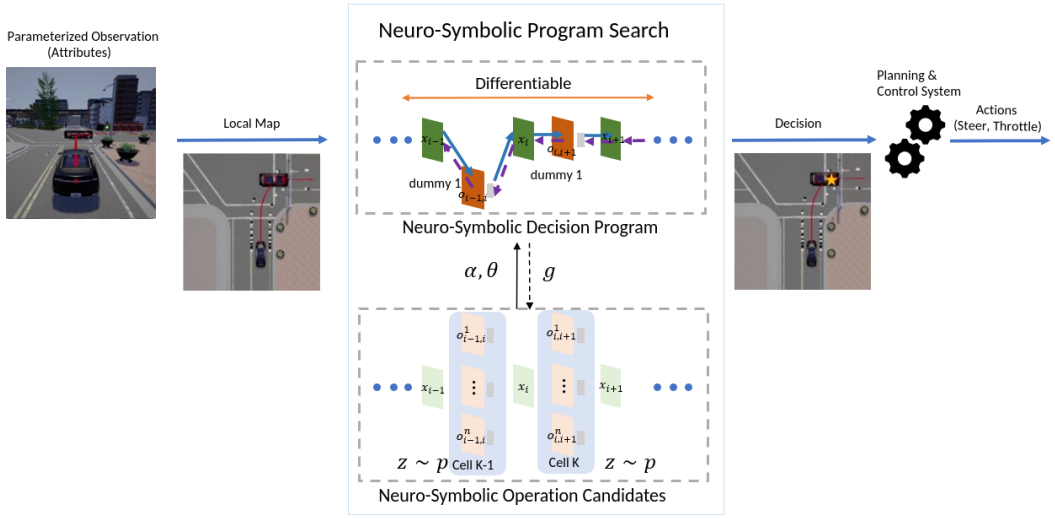


Figure 2: **NSPS Framework.** The Neuro-Symbolic Decision Program (NSDP) is searched by the Neuro-Symbolic Program Search (NSPS) through back-propagation g to update $\boldsymbol{\alpha}$, $\boldsymbol{\theta}$. The inputs of the modular pipeline of the autonomous driving system are parameterized observations. The decision policy, which is generated by NSDP, can be further used for planning and control. In the Neuro-Symbolic Decision Program, orange lumps are the selected operations $o_{i,j}$ and green lumps represent feature x_i generated by the selected operation. Blue arrow shows the direction of the forward data flow and purple dashed arrow indicates the backward data flow. The lower block in semi-transparent lumps shows the candidates for parent operations used in the search.

perception, decision, planning, and control. For the decision-making component in autonomous driving, finite state machine (FSM) is often adopted [15]. Building a complete rule-based FSMs system for decision making is infeasible. One issue for using FSM is that it is difficult to manage the implementation of FSM systems under a huge amount of manual rules. Recently, the end-to-end neural policy learned from expert driver's data using imitation learning is proposed based on generative adversarial learning [16, 17, 18]. Compared to the previous work above, our work aims at introducing end-to-end differentiable neuro-symbolic decision components to replace the pure neural network decision components in the autonomous driving pipeline, making the whole decision-making process more interpretable and transparent. With NSPS, we just need to define the neuro-symbolic operations. NSPS will automatically synthesize them into a program and tune the program hyperparameters to optimal.

Neural Architecture Search To automate the neural architecture design, Neural Architecture Search (NAS) [19, 20] comes up with various automatic search strategies to enumerate the network architecture spaces and obtain the optimal architecture under a certain task. Recent differentiable architecture search [21] has demonstrated its strength by modeling NAS as a single training process of a parent network that comprises all the candidate models. To mitigate the gap between the searching and evaluation stage in the existing framework, DSNAS [22] proposes a well-defined task-specific end-to-end NAS problem and applies a discrete solution to this problem. This enables the search of logical operations in NSPS to be possible. Different from previous NAS, the search space of NSPS is made of neuro-symbolic operations (numerical operations and logical operations) and some hyperparameters in NSP, instead of neural network operations.

3 Neuro-Symbolic Driving

Neuro-Symbolic Driving System learns to drive using the optimal neuro-symbolic decision program (NSDP) searched by NSPS. Figure 2 shows its framework. The inputs of the modular autonomous driving pipeline are the parameterized observations (a.k.a "attributes" in DSL, cf. Appendix). NSDP makes decisions to be passed to the downstream planning and control module. In the following subsections, we first introduce some background of DSNAS, which provide an algorithm guarantee for our method. Then we describe how the NSPS framework is formulated. Finally, we describe how

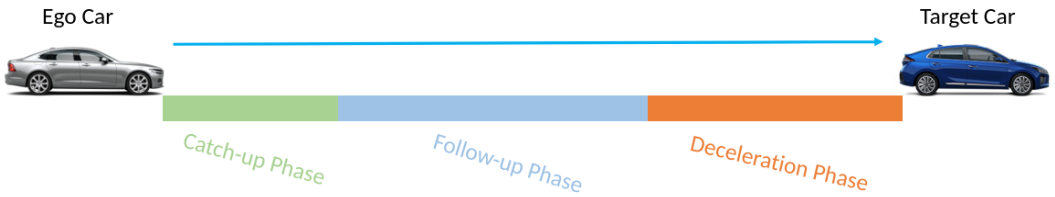


Figure 3: **The illustration of three phases.** Based on the distance between target car and ego car, their velocities and accelerations, there are three phases: Catch-up Phase, Follow-up Phase, and Deceleration Phase

NSPS is integrated into imitation learning algorithm GAIL [6] by reformulating the optimization framework.

3.1 Preliminary on Discrete Stochastic NAS

With the advantage of *plug-and-play*, DSNAS [22] can provide a *ready-to-deploy* network with optimized architecture and parameters, from which we derive our driving program search method. Thus we first provide a brief introduction of DSNAS.

As shown in Fig. 2, each node x_i (green lumps) in the directed acyclic graph (DAG) is an output of an operation. Each directed edge (i,j) (arrow lines) represents the flow of information between node x_i and node x_j. $p_{\boldsymbol{\alpha}}(Z)$, a categorical distribution parameterized by $\boldsymbol{\alpha}$, is adopted to represent the probability for N candidate operations. During the forward pass, a one-hot random variable $Z_{i,j}$ is first sampled from $p_{\boldsymbol{\alpha}}(Z)$ and then the sampled vector $Z_{i,j}$ is multiplied with the candidate operation $\boldsymbol{O}_{i,j}$.

$$\tilde{\boldsymbol{O}}_{i,j}(\cdot) = \boldsymbol{Z}_{i,j}^T \boldsymbol{O}_{i,j}(\cdot). \tag{1}$$

With architecture distribution parameters $\boldsymbol{\alpha}$ and neural operation parameters $\boldsymbol{\theta}$, the generic loss can be calculated by the Monte-Carlo sampling:

$$\mathbb{E}_{\boldsymbol{Z}\sim p_{\boldsymbol{\alpha}}(\boldsymbol{Z})}[L_{\boldsymbol{\theta}}(\boldsymbol{Z})]. \tag{2}$$

The one-hot random variable $\boldsymbol{Z}$ is relaxed to be a continuous random variable $\tilde{\boldsymbol{Z}}$ with the gumbel-softmax trick [23]. Policy gradient method can be adopted to estimate gradient:

$$\lim_{\lambda\to 0} \mathbb{E}_{\tilde{\boldsymbol{Z}}\sim p(\tilde{\boldsymbol{Z}})}\left[\frac{\partial L}{\partial \alpha_{i,j}^n}\right] = \mathbb{E}_{\boldsymbol{Z}\sim p(\boldsymbol{Z})}\left[\nabla_{\alpha_{i,j}^n} \log p(\boldsymbol{Z}_{i,j}) \left[\frac{\partial L}{\partial x_j}\sum_n O_{i,j}^n(x_i) Z_{i,j}^n\right]_c\right], \tag{3}$$

where λ is the temperature in gumble-softmax trick, $\tilde{\boldsymbol{Z}}_{i,j}$ is the gumbel-softmax random variable, $[\cdot]_c$ denotes that $\cdot$ is a *cost* independent from $\boldsymbol{\alpha}$ for gradient calculation, $\boldsymbol{Z}_{i,j}$ is a strictly one-hot random variable, $Z_{i,j}^n$ is the nth element.

Due to the fact that $\boldsymbol{Z}_{i,j}$ is one-hot random variable, *i.e.* only $Z_{i,j}^r$ on edge (i,j) is 1 while others are 0 (symbol "r" is the index of the value 1 in the sampled one-hot vector $Z_{i,j}$), the *cost* function can be reduced to

$$\frac{\partial L}{\partial x_j}\sum_n O_{i,j}^n(x_i) Z_{i,j}^n = \sum_n \frac{\partial L}{\partial x_j} O_{i,j}^n(x_i) Z_{i,j}^n = \frac{\partial L}{\partial x_j^i}\frac{\partial x_j^i}{\partial Z_{i,j}^r} = \frac{\partial L}{\partial Z_{i,j}^r}. \tag{4}$$

3.2 Neuro-Symbolic Program Search (NSPS)

In the autonomous driving pipeline of perception, decision, planning, and control, the decision module plays a crucial role in guiding the vehicle. We would like to design decision modules better and more efficiently. Therefore we propose Neuro-Symbolic Program Search (NSPS). The primary purpose of NSPS is to efficiently integrate such Neuro-Symbolic Decision Program (NSDP) by composing the symbolic operations.

Multiple neural symbolic operations $\boldsymbol{O}$ for different driving scenarios are designed. These operations are divided into two categories: logical operations and numerical operations. The logical operations such as `DecelerationPhase()`, `FollowUpPhase()`, and `CatchUpPhase()` can reflect which phase (cf. Figure 3) the vehicle is currently in. The numerical operations (*e.g.,*

Algorithm 1 NSPS with GAIL

Require: Expert trajectories $\tau_E \sim \pi_E$, parent program, policy parameters $\boldsymbol{\theta}$, discriminator parameters $\boldsymbol{w}$ and categorical architecture distribution $p_{\boldsymbol{\alpha}}(\boldsymbol{Z})$

$\boldsymbol{\theta}$, $\boldsymbol{\alpha} \leftarrow$ Initialized parameters

while not converged **do**

Sample one-hot random variables $\boldsymbol{Z} \sim p_{\boldsymbol{\alpha}}(\boldsymbol{Z})$

Construct child program as policy $\pi_{\boldsymbol{\theta}}$ with $\boldsymbol{\theta}$ according to $\boldsymbol{Z}$

Sample trajectories $\tau_G \sim \pi_{\boldsymbol{\theta}}$ and update the discriminator parameters from $\boldsymbol{w}$ with the gradient

$$\mathbb{E}_{\tau_G}[\nabla_{\boldsymbol{w}} \log D_{\boldsymbol{w}}(s,a)] + \mathbb{E}_{\tau_E}[\nabla_{\boldsymbol{w}} \log(1 - D_{\boldsymbol{w}}(s,a))] \tag{5}$$

Backward $\log(D_{\boldsymbol{w}}(s,a))$ to $\boldsymbol{\theta}$ by taking a KL-constrained natural gradient step with

$$\begin{aligned} \nabla_\theta \mathbb{E}_\pi[\log D_{\boldsymbol{w}}(s,a)] &\cong \hat{\mathbb{E}}_{\tau_G}[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)] \\ \textit{where } Q(\hat{s},\hat{a}) &= \hat{\mathbb{E}}_{\tau_G}[\log D_{\boldsymbol{w}}(s,a) | s_0 = \hat{s}, a_0 = \hat{a}] \end{aligned} \tag{6}$$

Calculate the gradient of $\boldsymbol{\alpha}$ using Eq. 3

Update program parameter $\boldsymbol{\theta}$ and architecture distribution parameter $\boldsymbol{\alpha}$

end while

`Intersect()`, `Union()`) are designed for numerical calculation. The details of these operations are listed in Appendix. They constitute a complete set of neural symbols for autonomous driving task and is enough to handle most scenarios. Note that all the neuro-symbolic operations are differentiable, which guarantees the end-to-end training of the neuro-symbolic program.

Inspired by DSNAS, which combines the efficiency of discrete sampling and the robustness of continuous differentiation, we formulate the Neuro-Symbolic Program Search as a stochastic optimization problem to integrate the candidate neuro-symbolic operations automatically. First, we give a formal definition of the terms used in our method. $\boldsymbol{O}$ in NSPS is defined as the neuro-symbolic operation set. As shown in Fig. 2, each cell consists of N operation candidates. Each node x_i (green lumps) in the DAG is an output of a neuro-symbolic operation. Each directed edge (i, j) (arrow lines) represents the flow of information between node x_i and node x_j, on which computation are performed on N candidate neuro-symbolic operations $\boldsymbol{O}_{i,j}$.

By multiplying the sampled one-hot random variable $Z_{i,j}$ from $p_{\boldsymbol{\alpha}}(Z)$ with the candidate neuro-symbolic operations $\boldsymbol{O}_{i,j}$ following Eq. 1, NSPS synthesizes an executable Neuro-Symbolic Decision Program (NSDP) represented in a domain-specific language (DSL). The DSL covers a set of fundamental operations $\boldsymbol{O}$ for autonomous driving. NSDP is a collection of deterministic functional modules. Given the searched NSDP, the driving decisions are derived based on structured input. To make the decision-making policy generalizable to different driving scenarios, the structured inputs include the 3D coordinates of waypoint of the ego vehicle's current lane. Lane points are sampled with exponentially increasing slots towards the further end, which mimics the effect of Lidar. Observation also includes the coordinates, accelerations, and velocities of the 6 nearest vehicles in traffic within the 70m range of the ego vehicle.

Our goal is to find the optimal neuro-symbolic program parameters $\boldsymbol{\theta} = \{\boldsymbol{\theta}_{c_1}, \boldsymbol{\theta}_{c_2}\}$ (including the distance threshold hyper-parameters of different phases, which are handcraft in previous methods) and architecture distribution parameters $\boldsymbol{\alpha}$ of the neuro-symbolic program, by minimizing the loss of Eq. 2 using DSNAS approach. As shown in Eq. 4, only one operation is selected on edge (i, j) during the searching process. This enables the search of logical operations in NSPS to be possible.

3.3 Learning from demonstration by integrating NSPS in GAIL

Autonomous vehicles should not only be safe but also provide a comfortable user experience. A large number of factors (e.g., jerk, acceleration, distances to other cars, speed during lane changes), characterize the comfort level of driving behavior. Due to the fact that defining a fixed and known reward or objective function for the desired behavior can be challenging and tedious for autonomous driving [5], we integrate NSPS into GAIL [6] and learn from demonstration (LfD) for autonomous driving as shown in Algorithm 1.

To update decision program parameters jointly, one way to take appropriate program search step size is to put a constraint on the KL divergence between the new policy and the old policy, i.e., a trust-region constraint [24]. In order to update neuro-symbolic program parameters $\boldsymbol{\theta}$ and the architecture parameters $\boldsymbol{\alpha}$ jointly, we reformulate the optimization problem,

$$\begin{aligned} \text{minimize} \quad & \mathbb{E}_{\boldsymbol{Z}\sim p_{\boldsymbol{\alpha}}(\boldsymbol{Z})}[L_{\boldsymbol{\theta}}(\boldsymbol{Z})] \\ \text{subject to} \quad & D_{KL}(\pi_{\boldsymbol{\theta}_{old}}(\cdot|s)||\pi_{\boldsymbol{\theta}}(\cdot|s)) \leq \delta_1, \\ & D_{KL}(P_{\boldsymbol{\alpha}_{old}}(\boldsymbol{Z})||P_{\boldsymbol{\alpha}}(\boldsymbol{Z})) \leq \delta_2, \end{aligned} \tag{7}$$

where $\pi_{\boldsymbol{\theta}}(\cdot|s)$ is the neuro-symbolic program, architecture random variable $\boldsymbol{Z}$ follows the categorical distribution $P_{\boldsymbol{\alpha}}(\boldsymbol{Z})$, $D_{KL}(\cdot||\cdot)$ is the KL divergence between two distributions, δ_1 and δ_2 are the bounds on KL divergence. In practice, $\log(D(s,a))$ can be used as objective function to update program parameters, where D is the discriminator in GAIL.

Note that different from the vanilla pipeline of GAIL [6], the generative process of our framework is the end-to-end differentiable Neuro-Symbolic Decision Program (NSDP) searched by NSPS. NSDP is differentiable and interpretable. With the provided expert data, NSDP can be adversarially trained with the discriminator in GAIL paradigm. The discriminator tries to distinguish between (s,a) of NSDP π and expert π_E. As an end-to-end imitation learning algorithm for autonomous driving, NSDP takes the structured observations s such as `Waypoint, Velocity, Acceleration, Bounding Box` as input and outputs driving decisions (target waypoint index, target velocity). Finally, the motion planner and controller convert the decisions to executable actions such as `Steer` and `Throttle`.

4 Experiments

4.1 Experimental Setup

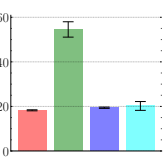
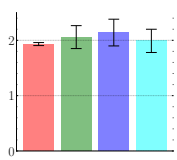
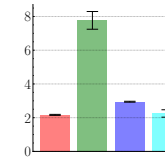
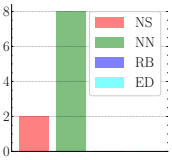
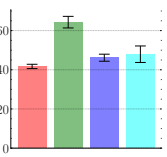
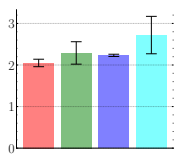
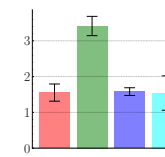
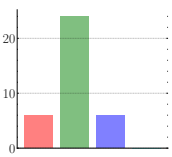
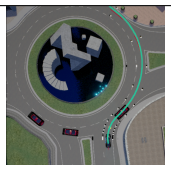
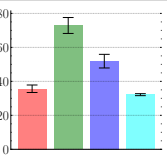
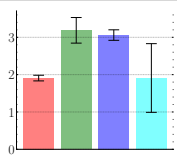
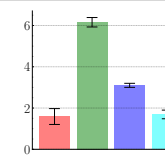
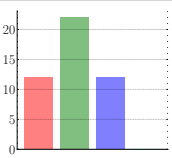
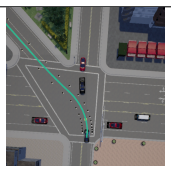
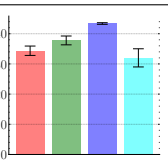
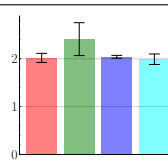
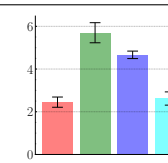
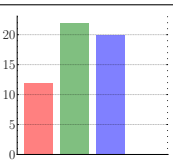
System Overview Neuro-Symbolic Driver (NSD) is developed based on an autonomous driving simulation platform CARLA [25], and is used for the experiments to collect expert data and evaluate the performance of Neuro-Symbolic Decision Programs. In this experiment, human participated in a driving data collection consisting of multiple scenarios. To keep things tight, we focus on the decision system. The parameterized observations are structured data provided by Neuro-Symbolic Driver. The differentiable Neuro-Symbolic Decision Program uses the ego-centric observations `Waypoint, Velocity, Acceleration, Bounding Box` as input and outputs driving decisions about target waypoint and target velocity. Executable actions such as `Throttle, Steer` are finally produced by the motion planner and controller based on these driving decisions. Note that even though `Vehicle Pose` is provided by Neuro-Symbolic Driver, it is not used as input since it is the global information. Please refer to Appendix for experiment details (scenario description, expert data, etc.).

Evaluation Metrics There are many different vehicle models (*e.g.*, Audi TT, Dodge, Etron, Lincoln, etc.) provided by CARLA. To evaluate the performance of the obtained program, we follow [26] to use metrics such as collision rate, time to accomplish tasks, average acceleration, and average jerk to conduct a quantitative comparison, which reflects how safe and smooth the agent drives. Table 1 presents the means and standard deviations of imitation learning performance, over the 50 trials. The jerk metric, as the temporal derivative of acceleration, is a quantity to measure driving comfort.

Search Space There are 12 searchable neuro-symbolic cells, which are each composed of $N = 12$ neuro-symbolic operations to form a neuro-symbolic program search space. Note that different from neural architecture search, the cells are not the same or stacked for multiple times. The input node of the first cell, fixed as `Filter()` operation, is used for processing the input observations (`Waypoint, Velocity, Acceleration, Bounding Box`). The output is the result of the selected operation in that cell, passed to the next searchable cells one by one in a sequence. Finally, the result of the last searchable cell is passed to a fixed operation `Query()` to generate the final action.

To improve the automation level of the system, besides the operations, our search space contains 2 searchable NSDP hyperparameters, the minimal safety distance coefficient $\boldsymbol{\theta}_{c_1}$ (the boundary

Table 1: Comparison of different methods. Columns indicate the scenario illustration, average time taken, acceleration, jerk, collision rate. (NS: Neuro-Symbolic Decision Program, NN: Neural Network, RB: Rule-based, ED: Expert Data)

Scenarios	Time taken (s)	Accel. (m/s^2)	Jerk (m/s^3)	Collision Rate (%)
Car Following					
Crossroad Merge					
Roundabout Merge					
Crossroad Turn Left					

between Deceleration Phase and Follow-up Phase) and the maximal safety distance coefficient $\boldsymbol{\theta}_{c_2}$ (the boundary between Follow-up Phase and Catch-up Phase), which previously handcraft in rule-based systems. In this way, our method can search not only the structure of the program but also the hyperparameters of the program. Note that the search procedure and the whole neuro-symbolic program are both end-to-end differentiable.

Compared Method Rule-based method and end-to-end neural policy are included as baselines. (1) Rule-based method is set up following [27] (more details in Appendix), which depends on lane-based coordinate and combines longitudinal and lateral proper responses. In the rule-based module, $\boldsymbol{\theta}_{c_1} = 1.5$ and $\boldsymbol{\theta}_{c_2} = 2.0$ is manually tuned. (2) The end-to-end policy is trained with vanilla GAIL [6] for 500 iterations (same with our method), which adopts neural network policy as a generative process. Nothing has changed except that the state space and action space have been adapted to the CARLA environment.

4.2 Results and Analysis

Quantitative Comparison The quantitative comparison between the performance of baseline methods and the Neuro-Symbolic Decision Program (NSDP) for autonomous driving obtained by our Neuro-Symbolic Program Search method is in Table 1 in terms of collision rate, time taken, average acceleration and average jerk. The driving behavior of the end-to-end policy would be unstable since it lacks the prior knowledge (*e.g.*, waypoints) to constrain its output. In practical application, the rule-based method requires manual architecture design and hyperparameter tuning (*e.g.*, $\boldsymbol{\theta}_{c_1}$ and $\boldsymbol{\theta}_{c_2}$). NSPS is able to replace the manual designing of rules, which potentially accelerates the process of automated programming. It can be seen that NSDP can generate relatively smoother driving behavior (lower acceleration and lower jerk) than the neural-network-based method, as measured by acceleration and jerk, which indicts passenger comfort. NSDP even achieves higher comfort (lower acceleration) than experts (cf. Appendix) in some scenarios due to human errors, and NSDP agent somehow fixes it with a more suitable program structure and hyperparameters. The program parameters of NSDP are also learnable. Using NSPS, hyperparameters $\boldsymbol{\theta}_{c_1} = 1.87$ and $\boldsymbol{\theta}_{c_2} = 2.16$ is searched for NSDP, which are different from handcraft values but the change is within the rea-

Table 2: Result of generalization to multiple scenarios

Scenarios	Neuro-Symbolic Decision Program			
	Collision Rate (%)	**Time taken** (s)	**Accel.** (m/s^2)	**Jerk** (m/s^3)
Car Following	4.00	19.21 ± 1.53	1.99 ± 0.21	2.25 ± 0.21
Crossroad Merge	12.00	47.95 ± 5.61	2.72 ± 0.31	1.54 ± 0.19
Roundabout Merge	10.00	36.23 ± 4.29	2.11 ± 0.14	1.69 ± 0.12
Crossroad Turn Left (Unseen)	20.00	34.91 ± 4.18	1.93 ± 0.15	2.45 ± 0.24

Table 3: Result of generalization to multiple speeds under Roundabout Merge scenario

Scenarios	Speed (km/h)	Neuro-Symbolic Decision Program			
		Collision Rate (%)	**Time taken** (s)	**Accel.** (m/s^2)	**Jerk** (m/s^3)
Roundabout Merge	21.00 ± 1.00	10.00	36.99 ± 5.21	2.21 ± 0.10	1.72 ± 0.21
	25.00 ± 1.00	10.00	39.56 ± 8.21	2.22 ± 0.19	1.70 ± 0.11
	35.00 ± 1.00 (Unseen)	12.00	37.15 ± 2.96	2.31 ± 0.11	1.72 ± 0.10

sonable range. The results of this automated search also save manual tuning time and help produce driving behavior superior to pure rule-based systems.

Generalization Generalization is a fundamental issue in neural networks based methods.We evaluate the generalization of our method in different scenarios. The training setting for multiple scenarios are as follows: actors collect data in simulators in various scenarios (*Car Following, Crossroad Merge*, and *Roundabout Merge*) simultaneously and run in parallel, while only one learner is deployed to learn a policy from all data sampled by actors and the provided expert demonstration. After the policy is learned, 50 trials are evaluated respectively on different (Seen & Unseen) scenarios using the learned policy.

We are also interested in the capability of agents to generalize at multiple speeds. Take *Roundabout Merge* scenario as an example, actors collect data in simulators at various speeds ($21km/h, 25km/h$), while only one learner is deployed to learn a policy. After the training is finished, 50 trials are evaluated on speeds $21km/h, 25km/h, 35km/h$ (Unseen) respectively, using the same learned policy.

The generalization results are shown in Tables 2 and 3. Compared to the Neuro-Symbolic Decision Program previously trained in each scenario separately as Table 1 shows, there is a slight drop in performance, with regard to metrics such as collision rate, time taken, acceleration, and jerk, but we can still say it holds up well. It can be seen that the policy with neuro-symbolic representation exhibits the generalization ability, and it can drive safely and smoothly (low collision rate, low acceleration, and low jerk) in scenarios of different complexity levels.

5 Conclusion

In this work, we propose a Neuro-Symbolic Program Search (NSPS) method for autonomous driving system design. NSPS framework searches for differentiable neuro-symbolic programs as well as corresponding program hyperparameters, which are previously manually crafted. Thus NSPS improves the design efficiency of neuro-symbolic programs over the previous rule-based system design. In addition to being able to cope with common traffic scenarios, our Neuro-Symbolic Decision Program brings more stable driving behavior than the neural-network-based and rule-based systems in terms of lower acceleration and jerk metrics. Our work can be seen as a step towards empowering autonomous driving with the neuro-symbolic program. Future work will be on the integration of the neuro-symbolic program into the processing of more complex autonomous driving scenarios.

Acknowledgement We thank Shoukang Hu, Sirui Xie, and Junning Huang for their valuable suggestions. We would also like to appreciate the insightful comments from anonymous reviewers.

References

[1] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1(1):33–55, 2016.

[2] U. Muller, J. Ben, E. Cosatto, B. Flepp, and Y. L. Cun. Off-road obstacle avoidance through end-to-end learning. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 739–746. MIT Press, 2006. URL http://papers.nips.cc/paper/2847-off-road-obstacle-avoidance-through-end-to-end-learning.pdf.

[3] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, 2019.

[4] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018.

[5] M. Kuderer, S. Gulati, and W. Burgard. Learning driving styles for autonomous vehicles from demonstration. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2641–2646, 2015.

[6] J. Ho and S. Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, pages 4565–4573, 2016.

[7] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu. The Neuro-Symbolic Concept Learner: Interpreting Scenes, Words, and Sentences From Natural Supervision. In *International Conference on Learning Representations*, 2019.

[8] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[9] T. R. Besold, A. d. Garcez, S. Bader, H. Bowman, P. Domingos, P. Hitzler, K.-U. Kühnberger, L. C. Lamb, D. Lowd, P. M. V. Lima, et al. Neural-symbolic learning and reasoning: A survey and interpretation. *arXiv preprint arXiv:1711.03902*, 2017.

[10] L. P. Kaelbling and T. Lozano-Pérez. Hierarchical planning in the now. In *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[11] M. Garnelo, K. Arulkumaran, and M. Shanahan. Towards deep symbolic reinforcement learning. *arXiv preprint arXiv:1609.05518*, 2016.

[12] E. Parisotto, A.-r. Mohamed, R. Singh, L. Li, D. Zhou, and P. Kohli. Neuro-symbolic program synthesis. *International Conference on Learning Representations*, 2017.

[13] Z. Manna and R. J. Waldinger. Toward automatic program synthesis. *Communications of the ACM*, 14(3):151–165, 1971.

[14] M. Buehler, K. Iagnemma, and S. Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. springer, 2009.

[15] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, et al. Junior: The stanford entry in the urban challenge. *Journal of Field Robotics*, 25(9):569–597, 2008.

[16] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer. Imitating driver behavior with generative adversarial networks. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 204–211. IEEE, 2017.

[17] F. Behbahani, K. Shiarlis, X. Chen, V. Kurin, S. Kasewa, C. Stirbu, J. Gomes, S. Paul, F. A. Oliehoek, J. Messias, et al. Learning from demonstration in the wild. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 775–781. IEEE, 2019.

[18] J. Huang, S. Xie, J. Sun, Q. Ma, C. Liu, D. Lin, and B. Zhou. Learning a decision module by imitating driver's control behaviors. *arXiv preprint arXiv:1912.00191*, 2019.

[19] H. Liu, K. Simonyan, and Y. Yang. DARTS: Differentiable architecture search. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=S1eYHoC5FX.

[20] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean. Efficient neural architecture search via parameters sharing. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4095–4104, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/pham18a.html.

[21] S. Xie, H. Zheng, C. Liu, and L. Lin. SNAS: stochastic neural architecture search. In *International Conference on Learning Representations*, 2019.

[22] S. Hu, S. Xie, H. Zheng, C. Liu, J. Shi, X. Liu, and D. Lin. Dsnas: Direct neural architecture search without parameter retraining. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[23] C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables. *International Conference on Learning Representations*, 2017.

[24] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897, Lille, France, 07–09 Jul 2015. PMLR. URL http://proceedings.mlr.press/v37/schulman15.html.

[25] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[26] S. Bansal, V. Tolani, S. Gupta, J. Malik, and C. Tomlin. Combining optimal control and learning for visual navigation in novel environments. 2019.

[27] S. Shalev-Shwartz, S. Shammah, and A. Shashua. On a formal model of safe and scalable self-driving cars. *arXiv preprint arXiv:1708.06374*, 2017.