2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Boosting Offline Reinforcement Learning for Autonomous Driving with Hierarchical Latent Skills

Zenan Li[1,2], Fan Nie[2,3], Qiao Sun[2], Fang Da[4], and Hang Zhao[†,1,2]

***Abstract*— Learning-based vehicle planning is receiving increasing attention with the emergence of diverse driving simulators and large-scale driving datasets. While offline reinforcement learning (RL) is well suited for these safety-critical tasks, it still struggles to plan over extended periods. In this work, we present a skill-based framework that enhances offline RL to overcome the long-horizon vehicle planning challenge. Specifically, we design a variational autoencoder (VAE) to learn skills from offline demonstrations. To mitigate posterior collapse of common VAEs, we introduce a two-branch sequence encoder to capture both discrete options and continuous variations of the complex driving skills. The final policy treats learned skills as actions and can be trained by any off-the-shelf offline RL algorithms. This facilitates a shift in focus from per-step actions to temporally extended skills, thereby enabling long-term reasoning into the future. Extensive results on CARLA prove that our model consistently outperforms strong baselines at both training and new scenarios. Additional visualizations and experiments demonstrate the interpretability and transferability of extracted skills.**

I. Introduction

Learning-based vehicle planning is increasingly gaining its popularity with the advent of driving simulators [1, 2] and large-scale offline datasets [3, 4]. Among them, the performance of Imitation Learning (IL) algorithms is limited by the quality of expert data [5] as well as the covariate shift issue [6, 5], while Reinforcement Learning (RL) necessitates online interaction with the environment. In contrast, offline RL [7, 8, 9] emerges as an effective approach to learning policies that break the constraints of datasets while avoiding risky exploration in real driving environments.

Despite its great potential, the application of offline RL in autonomous driving is relatively limited [10, 11, 12], with the long-horizon problem presenting as a challenging obstacle [12, 13]. Specifically, the reward signals in long-horizon tasks are sparse and delayed [14, 15]. However, the agent must learn from them to plan a sequence of actions that can avoid compounding errors over extended periods.

One powerful strategy to deal with long-horizon tasks is to adopt a hierarchical framework [16, 12], with the high-level policy producing a subgoal, and the low-level policy executing to approach the consecutive subgoals. As pointed out by [17], human behaviors are fundamentally skill executions. Building upon this, our intuition is that the vehicle planning problem can also be solved by acquiring a set of *driving skills* (e.g., cruising, nudging, and turning) [13, 18].

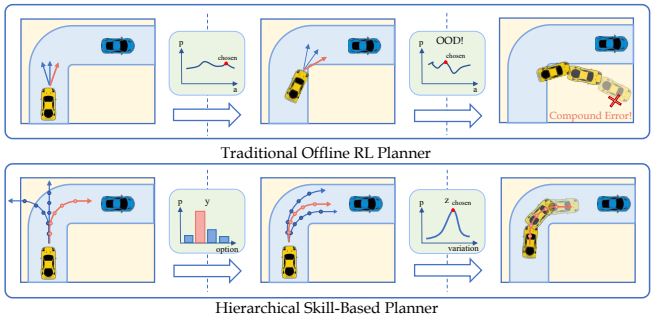


Fig. 1: A motivating example. Top: Traditional offline RL planner samples action at each timestep and is easy to go outside the training data distribution, thus incurring compounding errors of actions. Bottom: Our proposed hierarchical skill-based planner first selects the discrete skill option (turning), and then decides the continuous variation for execution (radians), which can model a diverse set of driving skills. This multi-step skill execution enables long-term reasoning for the planner.

Therefore, we propose to interpret the subgoals as different driving skills to decompose the long-horizon task into a two-stage problem: (i) extract a set of reusable driving skills, and (ii) learn a skill-based policy. Essentially, skills provide a way to represent a sequence of actions. This temporal abstraction [19] allows the vehicle to reason and plan at a higher level of granularity [20, 21]. It considers longer-term consequences (so densely shaping reward signals), thereby improving long-horizon planning.

Several prior works have investigated techniques for extracting skills from offline datasets [21, 22, 23]. Generally, they all employ a sequence-to-sequence conditional variational autoencoder (seq-to-seq CVAE) [24, 25] to model the generative process of transition sequences, and extract a continuous latent space of skills describing different driving behaviors. This straightforward method can, however, suffer in the vehicle planning setting since the expert demonstrations are seriously *unbalanced* in distribution, with a majority dominated by simple straight or static sequences. While this can be addressed by pre-filtering the dataset, a more fundamental challenge stems from the *complex and versatile* nature of driving skills. Specifically, drivers must react aptly to different maps and environmental vehicle configurations, which encompass an infinite number of combinations. For example, turning behaviors vary significantly across different maps and road conditions. Typically, VAEs are widely known to suffer from the posterior collapse problem [26, 27] under high input complexity. As shown in Fig. 4(a), the VAE encoder fails to model and distinguish between different driving skills.

The key to overcoming posterior collapse is to use more expressive latent distributions that can better capture diverse aspects of inputs [28]. Regarding this objective, we notice the

*Zenan Li and Fan Nie contribute equally to the paper.
[1]Tsinghua University, [2]Shanghai Qi Zhi Institute, [3]Shanghai Jiao Tong University, [4]QCraft Inc, [†]Corresponding author.
Emails in order: *li-zn23@mails.tsinghua.edu.cn*, *youluo2001@sjtu.edu.cn*, *alan.qiao.sun@gmail.com*, *fang@qcraft.ai*, *hangzhao@mail.tsinghua.edu.cn*.

common structure of driving skills in Fig. 1: Generally, drivers first select within discrete skill options (e.g., turning), then determine continuous variations (e.g., radians and speeds) during execution. Therefore, we suggest injecting this prior into the generative process and extracting a hierarchical latent space instead of simple Gaussian distributions. Specifically, our VAE encoder is two-branch: the discrete branch produces logits that are passed through a Gumbel-softmax activation [29] to get approximately one-hot variables; the continuous branch functions as an ensemble, with discrete outputs acting as gates to select a member network for computing the final skill variable. With the discrete branch distinguishing between different skill options, the member networks in the continuous ensemble only need to handle variations within a specific skill option. This leads to a comparatively simple distribution, which helps to reduce the risk of posterior collapse.

After distilling skills from data, any off-the-shelf offline RL algorithm [8, 30, 31] can be employed to learn the high-level skill-based policy using relabeled transition data. The VAE decoder is utilized as the low-level action decoder, which can also be finetuned with conditional behavior cloning (BC).

To summarize, we make the following contributions:

- We present **H**ierarchical **s**kill-based **O**ffline Reinforcement Learning for **V**ehicle **P**lanning (HsO-VP), a framework that boosts offline RL to solve the challenging long-horizon vehicle planning task.
- We propose a two-branch network architecture for the VAE encoder, thus forming a hierarchical latent space that can effectively address the posterior collapse issue during driving skill extraction.
- We comprehensively evaluate HsO-VP on CARLA [1], where it achieves consistent performance improvements over the baselines (4.0% at training scenarios and 6.4% at new scenarios). Besides, the learned skills are also verified to possess interpretability and transferability.

II. Related Works

Vehicle planning aims to efficiently drive the ego-vehicle to the destination while conforming to some safety and comfort constraints, which is essentially a sequential decision problem. Recently, learning-based planning algorithms [32, 33, 34, 35] are gaining popularity over conventional rule-based algorithms [36, 37] for their better scalability to different driving scenarios. In this section, we review previous works about offline RL and skill-based algorithms as foundations of our work, and highlight the differences and contributions of HsO-VP.

Offline RL: Offline RL algorithms learn policies from a static dataset without interacting with the real environment [7], which is particularly useful in scenarios where interaction can be prohibitively expensive or risky, like autonomous driving. Unlike IL, offline RL algorithms have been proven to be capable of surpassing the performance of experts [8, 30, 31]. Generally, they utilize policy regularization [8, 38] and out-of-distribution (OOD) penalization [30, 39] to avoid value overestimation. In this paper, we pioneer to deploy offline RL algorithms to the long-horizon vehicle planning task, where the reward may be sparse and the extrapolation error will be accumulated [14, 15]. Although there have been several works [10, 11, 12] attempting to apply offline RL to vehicle planning, few of them [12] directly deal with the long-horizon challenge. As the most relevant work to our paper, HiGoC [12] also employs a hierarchical planning framework (not open-sourced, so we choose IRIS [40] instead as the comparing baseline). Its high-level policy produces subgoal states through optimization, and the low-level policy is trained by offline RL conditioned on subgoals. In contrast, HsO-VP utilizes skill completion as subgoals instead of imagined target states, which offers better interpretability in driving tasks. Besides, offline RL is conducted on the high-level skill-based policy in HsO-VP, enabling long-term reasoning during training.

Skill-based algorithms: Skills, also known as primitives or macro actions, are widely used in RL to enable knowledge transfer on new tasks [21, 41] or facilitate long-horizon training [23, 42]. Previous works [18, 43] have attempted to design driving skills for specific tasks manually. While these task-specific skill designs can succeed in certain well-defined tasks, such as overtaking a car, they limit the flexibility and expressiveness in complex driving scenarios [13]. In addition, another line of research explores skills from the perspective of unsupervised segmentation of reusable behaviors in temporal data [21, 22, 23, 41, 42]. Generally, they use a VAE to extract continuous skill representations from expert transition sequences. Once the skills have been extracted, a new high-level skill-based policy is learned from relabeled transitions by online [21, 22, 41] or offline RL [23, 42]. The VAE decoder acts as the low-level policy that outputs per-step actions conditioned on high-level skills. In this paper, we find vanilla VAEs not expressive enough for modeling the versatile driving demonstrations. As a treatment, we introduce another categorical latent variable to form a hierarchical latent space that can capture both discrete options and continuous variations of driving skills, enhancing the interpretability and transferability of learned skills. Different from [13] that learns driving skills from sampled trajectories, we extract interpretable skills from expert demonstrations and focus on the offline RL setting.

III. Preliminary

We introduce preliminary knowledge for HsO-VP in this section. To keep notations concise, we use subscripts t, c or numbers for variables at specific timesteps, Greek letter subscripts for parameterized variables, and bold symbols to denote variables spanning multiple timesteps.

We consider learning in a Markov decision process (MDP) [44] denoted by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state space and the action space, respectively. Given states $s, s' \in \mathcal{S}$ and action $a \in \mathcal{A}$, $\mathcal{P}(s'|s,a) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ is the transition probability distribution and $r(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ defines the reward function. Besides, $\gamma \in (0,1]$ is the discount factor. The agent takes action a at state s according to its policy $\pi(a|s) : \mathcal{S} \times \mathcal{A} \to [0,1]$. The goal of *online RL* is to find a policy π that maximizes the total expected return:

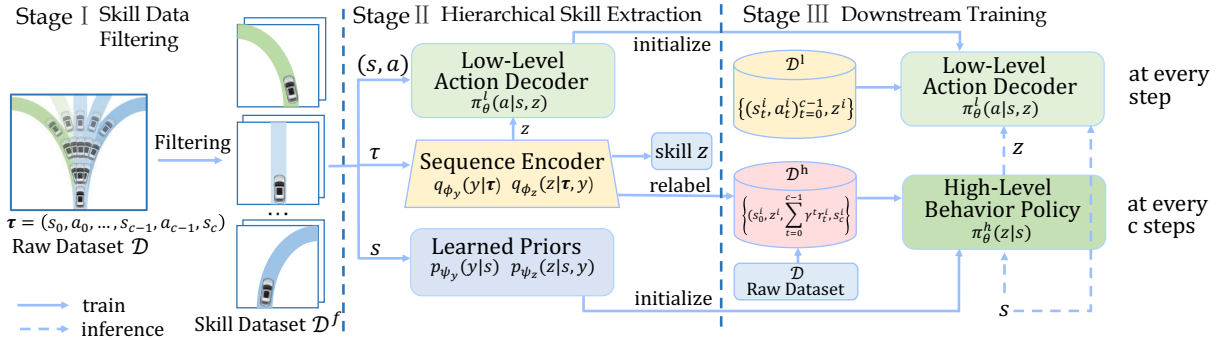


Fig. 2: Overview of HsO-VP. Left: Pre-filter the raw dataset to get the balanced skill dataset. Middle: Main components of our proposed VAE with hierarchical latent space for driving skill extraction. Right: Downstream training for our skill-based policy. Both high- and low-level policies are initialized from Stage II and trained by offline RL and conditional BC from the relabeled datasets, respectively.

$J = \mathbb{E}_{a_t\sim\pi(\cdot|s_t),s_{t+1}\sim\mathcal{P}(\cdot|s_t,a_t)}\left[\sum_{t=0}^{T-1}\gamma^t r(s_t,a_t)\right]$ by learning from the transitions (s, a, r, s') through interacting with the environment in an online manner. In contrast, *Offline RL* avoids safety issues during certain online interaction scenarios. It makes use of a static dataset with N trajectories $\mathcal{D} = \{\tau_i\}_{i=1}^N$ collected by a behavior policy π_b to learn a policy π that maximizes J. Here $\tau = \left\{(s_t, a_t, r_t, s_t')\right\}_{t=0}^{T-1}$ is a collected interaction trajectory composed of transitions with horizon T.

IV. Approach: HsO-VP

In this section, we begin with an overview of HsO-VP's insight and architecture, followed by detailed explanations of composition modules used in the approach.

A. Model Overview

An overview of the proposed approach HsO-VP is illustrated in Fig. 2. Specifically, it consists of three stages: First, we conduct *skill data filtering* for a balanced skill dataset, facilitating subsequent VAE training. Second, a two-branch VAE is specially designed for *hierarchical skill extraction*, modeling structured and complex skill representations to overcome the posterior collapse problem. Finally, the extracted skills (functioning as temporal abstractions to enable long-horizon planning) are used to relabel the dataset for *downstream training*. In the following, we introduce each stage with more details. For convenience, we use the terms 'skill' and 'behavior' interchangeably.

B. Skill Data Filtering

The temporally extended skills can be a solution to boost offline RL for long-horizon vehicle planning, which enables the vehicle to plan at a higher level of granularity and facilitates long-term reasoning into the future [20, 21]. However, raw expert demonstrations are not well-suited for skill extraction due to their unbalanced distribution (with straight and static trajectories occupying the majority). Therefore, we propose to pre-filter the transition sequences $\boldsymbol{\tau} = (s_0, a_0, ..., s_{c-1}, a_{c-1}, s_c)$, where c stands for the skill length, for a balanced filtered skill dataset $\mathcal{D}^f$. As an auxiliary preprocessing stage, meticulous data management is not necessary. Given this, and since it's not the primary focus of this work, here we briefly introduce our filtering principle.

Since the action sequences can already reflect the driving skills to a great extent (e.g., the turning and straight behaviors can be easily distinguished by corresponding action sequences), we concatenate these c-step low-dimensional actions into vectors $\boldsymbol{a}_{0:c-1}$ and conduct clustering [45]. With different clusters coarsely representing different skills, our objective is then to maintain uniformity in the sequence count within each cluster. In practical implementation, we control and balance sequence counts through a set of thresholds. Specifically, we choose a threshold for each cluster, filter and retain action sequences whose pairwise distances are greater than (different therefore representative) the threshold.

C. Hierarchical Skill Extraction

With the filtered balanced skill dataset, we can step to extract reusable driving skills through variational inference (VI) [25], similar to previous works [21, 22, 41].

Generative Process: The key of VI is to model the generative process of transition sequences and their corresponding skill variables z. Generally, it can be written as follows:

$$p(\boldsymbol{\tau}, z) = p(\boldsymbol{s}_{0:c}, \boldsymbol{a}_{0:c-1}, z) = p(s_0)p(z|s_0)\cdot \prod_{t=0}^{c-1} p(a_t|s_t, z)p(s_{t+1}|s_t, a_t). \tag{1}$$

In consideration of vehicle planning, however, the skills can be more complex and versatile than those in other fields [13, 18] across different maps and road conditions. This complexity can give rise to posterior collapse [26, 27], where the VAE fails to capture complex dependencies of input data and results in a degenerated indistinguishable latent space. To boost the expressive capability of modeling driving skills, we propose explicitly modeling the skill structure to better learn the latent representations. Driving behavior typically involves selecting a discrete skill option (e.g., turning) and then determining the specific variations (e.g., speed). To capture these discrete options, we introduce another categorical variable $y \in \mathbb{1}^K$ and construct a hierarchical latent space.

$$p(\boldsymbol{\tau}, y, z) = p(\boldsymbol{s}_{0:c}, \boldsymbol{a}_{0:c-1}, y, z) = p(s_0)p(y|s_0)\cdot p(z|y, s_0)\prod_{t=0}^{c-1} p(a_t|s_t, z)p(s_{t+1}|s_t, a_t). \tag{2}$$

Through this hierarchical decomposition, z is required to capture variations within a specific discrete option (depending on y) instead of the entire data distribution. This relatively simple distribution helps mitigate the risk of posterior collapse. Practically, we model the generative priors by some

flexible parametric distributions $p_{\psi_y}(y|s_0)p_{\psi_z}(z|y,s_0)$. To keep notations consistent with the RL setting, we use a low-level action decoder $\pi^l_\theta(a_t|s_t,z)$ to substitute for $p(a_t|s_t,z)$. Besides, the state transition distribution $p(s_{t+1}|s_t,a_t)$ is fixed and decided by the environment.

Learning Objective: To infer latent skills from data, we introduce transition sequence encoders $q_{\phi_y}(y|\boldsymbol{\tau})q_{\phi_z}(z|\boldsymbol{\tau},y)$ to approximate the intractable posterior $p(y|\boldsymbol{\tau})p(z|\boldsymbol{\tau},y)$. Based on these, the target of VI is to maximize the *conditional probability* $\log p_{\theta,\psi}(\boldsymbol{\tau}|s_0)$ of the observed transition sequence $\boldsymbol{\tau}$ w.r.t. the initial state s_0, which is deduced to be larger than the Evidence Lower BOund (ELBO) [25]. Our learning objective is therefore transformed to maximize the ELBO:

$$\max_{\phi,\psi,\theta} \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D}^f, y\sim q_{\phi_y}(y|\boldsymbol{\tau}), z\sim q_{\phi_z}(z|\boldsymbol{\tau},y)} \Bigg[\underbrace{\sum_{t=0}^{c-1} \log \pi^l_\theta(a_t|s_t,z)}_{\text{reconstruction loss } \mathcal{L}_{\text{recon}}} \Bigg]$$

$$-\beta_z \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D}^f, y\sim q_{\phi_y}(y|\boldsymbol{\tau})} \Bigg[\underbrace{D_{\text{KL}}\Big(q_{\phi_z}(z|\boldsymbol{\tau},y)||p_{\psi_z}(z|s_0,y)\Big)}_{\text{continuous regularization } \mathcal{L}^{\text{cont}}_{\text{KL}}} \Bigg]$$

$$-\beta_y \mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{D}^f} \Bigg[\underbrace{D_{\text{KL}}\Big(q_{\phi_y}(y|\boldsymbol{\tau})||p_{\psi_y}(y|s_0)\Big)}_{\text{discrete regularization } \mathcal{L}^{\text{dis}}_{\text{KL}}} \Bigg]. \tag{3}$$

For practical implementation, we adopt the β-VAE [46] trick and weigh the KL terms with β_y and β_z, respectively.

Network Instantiation: Finally, we specify the network architecture of the proposed modules for skill extraction. For all vehicle planners implemented in this paper, we use the same state input composed of a bird's-eye view (BEV) semantic segmentation image and a measurement vector.

- **[Sequence encoders]** $q_{\phi_y}(y|\boldsymbol{\tau})$ and $q_{\phi_z}(z|\boldsymbol{\tau},y)$: The architecture of our proposed sequence encoder is shown in Fig. 3. For the state representation, we use convolutional layers to encode the BEV and fully connected (FC) layers for the measurement vector. Afterward, a gated recurrent unit (GRU) [24] is adopted to process the c-step transition inputs. For modeling the driving skill structure, we use a two-branch architecture after the GRU: the discrete branch takes in the GRU outputs and produces K-dimensional logits for y; the continuous branch is composed of K-ensembled FC layers, which outputs final continuous skill variable z by weighted summation with y acting as coefficients. Notably, we use gumbel-softmax [29] activation to sample approximately one-hot discrete variables y, functioning as gates to guide members in the ensemble to learn about skills from different options.
- **[Low-level action decoder]** $\pi^l_\theta(a|s,z)$: We use a GRU decoder that takes in state and skill embeddings to output low-level executable actions.

The learnable priors $p_{\psi_y}(y|s)$ and $p_{\psi_z}(z|s,y)$ and the high-level behavior policy $\pi^h_\theta(z|s)$ just share the same architecture as sequence encoders except that we remove the GRU and directly use the single-step state representation for y and z.

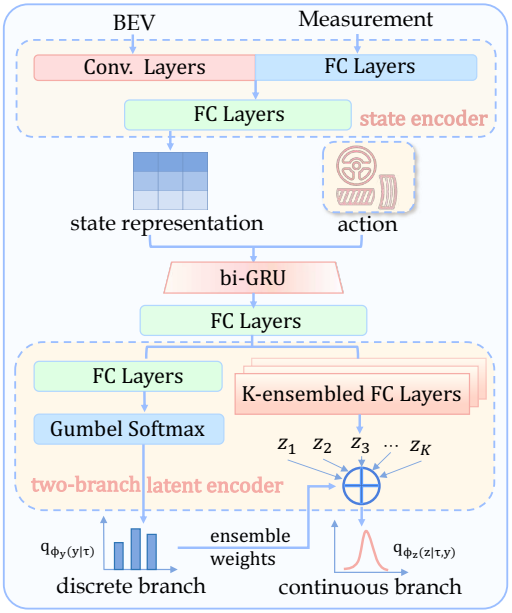


Fig. 3: The two-branch network architecture of our proposed transition sequence encoder. Note that the input to bi-GRU is actually composed of states and actions over several timesteps.

D. Downstream Training

Once the skills have been learned from $\mathcal{D}^f$ in terms of sequence encoders $q_{\phi_y}(y|\boldsymbol{\tau})q_{\phi_z}(z|\boldsymbol{\tau},y)$, action decoder $\pi^l_\theta(a|s,z)$, and priors $p_{\psi_y}(y|s)p_{\psi_z}(z|s,y)$, HsO-VP can then apply these modules to learn a skill-based vehicle planner.

Relabeled skill dataset: To learn a high-level skill-based policy $\pi^h_\theta(z|s)$ with offline RL, we need to relabel transitions in $\mathcal{D}$ (w/o filtering). Specifically, for each c-step sequence sampled from the dataset $\boldsymbol{\tau} \sim \mathcal{D}$, we label its corresponding skill by sequence encoders $z \sim \mathbb{E}_{y\sim q_{\phi_y}(y|\boldsymbol{\tau})} q_{\phi_z}(z|\boldsymbol{\tau},y)$, thereby creating a new dataset $\mathcal{D}^h = \{(s^i_0, z^i, \sum_{t=0}^{c-1}\gamma^t r^i_t, s^i_c)\}_{i=1}^{N^h}$ that treats skills as actions.

Downstream Learning Procedure: Given the relabeled dataset $\mathcal{D}^h$, any off-the-shelf offline RL algorithm can be used to learn $\pi^h_\theta(z|s)$. In this paper, we choose IQL [31], one of the state-of-the-art (SOTA) offline RL algorithms for end-to-end policy training. Besides, the high-level behavior policy is initialized with the learned priors $\mathbb{E}_{y\sim p_{\psi_y}(y|s)} p_{\psi_z}(z|s,y)$, which can be a good starting point to solve some short-horizon tasks.

During inference, $\pi^h_\theta(z|s)$ is invoked at every c-steps, while $\pi^l_\theta(a|s,z)$ is used to give executable low-level actions for the vehicle. To ensure that the c-step transitions $(s_t,a_t)_{t=0}^{c-1}$ remain consistent with the labeled latent skill z, we can also optionally finetune $\pi^l_\theta(a|s,z)$ on $\mathcal{D}^l = \{(s^i_t, a^i_t)_{t=0}^{c-1}, z^i\}_{i=1}^{N^l}$ with a skill-conditioned behavior cloning (BC) loss:

$$\min_\theta \mathbb{E}_{(\boldsymbol{\tau},z)\sim\mathcal{D}^l} \Bigg[-\sum_{t=0}^{c-1} \log \pi^l_\theta(a_t|s_t,z) \Bigg]. \tag{4}$$

V. Evaluation

In this section, we do extensive experiments to answer the following questions. **Q1:** How effective is HsO-VP compared to baselines when deployed in seen or unseen driving scenarios? **Q2:** How do the composed modules impact

HsO-VP's performance? **Q3:** Do skills extracted by HsO-VP possess interpretability and transferability?

A. Experiment Setup

Datasets: The training dataset $\mathcal{D}$ is collected in the CARLA simulator with an expert RL-based planner [34] trained with BEV semantic images and privileged vehicle measurements. More specifically, we collect data at 10Hz from 4 different training towns (Town01, Town03, Town04, Town06) and 4 weather conditions (ClearNoon, WetNoon, HardRainNoon, ClearSunset) for a total of 10 hours of driving data. At each timestep, we save a tuple (i_t, m_t, a_t, r_t), with $i_t \in \mathbb{R}^{15\times192\times192}$ the BEV semantic image, $m_t \in \mathbb{R}$ the velocity of the ego-vehicle, $a_t \in [-1, 1]^2$ the action for steering and acceleration executed by the expert, and $r_t \in \mathbb{R}$ the current step reward from the environment. Our reward design is modified from [34], decided by elements including speed, safety, comfort, etc. Besides, to study skill transferability, we also collect 5 hours of driving data from Town05.

Metrics: We report metrics from the CARLA challenge [1, 47] to measure planners' driving performance: infraction score, route completion, success rate, and driving score. Besides, as done in [35], we also report the normalized reward (the ratio of total return to the number of timesteps) to reflect the driving performance at the timestep level. Among them, driving score is the most significant metric for evaluating planners' performance, which is a weighted score of various indicators like driving efficiency, safety, and comfort.

Baselines: First, we choose two IL baselines: vanilla Behavior Cloning (BC) and Monotonic Advantage Re-Weighted Imitation Learning (MARWIL) [48]. Apart from IL baselines, we also include SOTA offline RL baselines: Conservative Q-Learning (CQL) [30], Implict Q-Learning (IQL) [31]. Finally, we add two SOTA hierarchical offline RL algorithms: Implicit Reinforcement without Interaction at Scale (IRIS) [40] that generates subgoals using offline RL, and Offline Primitives for Accelerating offline reinforcement Learning (OPAL) [23] which extracts skills using vanilla VAEs, as rigorous baselines.

Implementation Details: Training is conducted on 8 NVIDIA 3090 GPUs, while inference is performed on one NVIDIA 3090 GPU to ensure a fair comparison. For skill filtering and extraction, we remove background vehicles to prevent any detrimental impact on the extraction of the ego-vehicle's skills due to their stochastic movements [13]. These vehicles are retained in the downstream offline RL process for the ego-vehicle to make reasonable reactions. For filtering, the cluster number is set as 6 and we retain approximately 50% of expert data. The skill length c is set as 10 (corresponding to 1 second of driving). The skill extraction networks are trained from scratch using the Adam optimizer (with a learning rate of 0.0001) for 1000 epochs, employing a batch size of 64. The gumbel-softmax temperature is 0.1, and regularization factors β_y and β_z are set to 0.01. The number of discrete skill options K (i.e. dimension of y) is 6, the same as the cluster number. The downstream training process shares the same optimizer configuration with the skill extraction stage, and the discounted factor γ is set to 0.99.

Tab. I: Driving performance on a train town and train weather conditions in CARLA. Mean and standard deviation are computed over 3 evaluation seeds. See the main text for detailed meanings of HsO-VP variants. All metrics are recorded in percentages (%) except the normalized reward. The best results are in bold and our method is colored in gray.

Planner	Driving Score↑	Success Rate↑	Route Co.↑	Infraction Score↑	Norm. Reward↑
BC	65.7 ± 2.8	44.2 ± 6.2	78.3 ± 3.1	73.3 ± 1.8	0.40 ± 0.08
MARWIL [48]	66.8 ± 2.3	45.0 ± 7.1	80.7 ± 1.2	73.4 ± 1.2	0.46 ± 0.01
CQL [30]	69.4 ± 3.6	50.3 ± 8.5	81.5 ± 3.5	76.2 ± 3.0	0.56 ± 0.06
IQL [31]	70.7 ± 4.4	51.2 ± 6.6	82.0 ± 2.0	76.7 ± 5.0	0.57 ± 0.05
IRIS [40]	72.8 ± 2.8	54.0 ± 4.8	85.1 ± 2.6	78.3 ± 3.2	0.60 ± 0.06
OPAL [23]	72.1 ± 2.4	53.2 ± 5.5	84.3 ± 4.1	77.9 ± 2.4	0.59 ± 0.08
HsO-VP Raw	68.3 ± 2.4	47.3 ± 6.6	81.5 ± 3.5	74.9 ± 1.4	0.51 ± 0.04
HsO-VP w/o BC	74.1 ± 1.2	57.8 ± 4.4	83.3 ± 2.1	80.2 ± 2.2	0.62 ± 0.03
HsO-VP	**76.8 ± 1.4**	**59.2 ± 4.5**	**86.8 ± 3.3**	**82.4 ± 1.4**	**0.65 ± 0.03**
Expert [34]	78.7 ± 1.1	60.0 ± 2.0	89.5 ± 5.4	84.1 ± 1.2	0.69 ± 0.03

Tab. II: Driving performance on a new town and new weather conditions in CARLA. Transfer learning results are colored in green. The best results in bold do not take into account the transfer learning results.

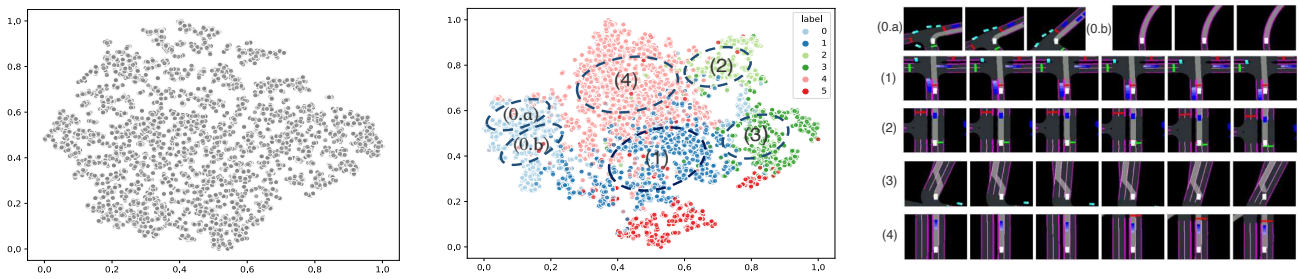
Planner	Driving Score↑	Success Rate↑	Route Co.↑	Infraction Score↑	Norm. Reward↑
BC	68.8 ± 2.5	48.3 ± 6.2	80.0 ± 7.1	74.7 ± 0.8	0.35 ± 0.17
MARWIL [48]	70.9 ± 6.7	58.1 ± 9.5	83.3 ± 2.4	76.0 ± 6.6	0.41 ± 0.13
CQL [30]	72.8 ± 1.6	61.3 ± 6.2	85.1 ± 3.1	77.1 ± 3.0	0.54 ± 0.01
IQL [31]	73.4 ± 3.8	63.0 ± 4.1	86.3 ± 3.1	77.8 ± 4.8	0.56 ± 0.03
IRIS [40]	76.4 ± 3.3	65.8 ± 5.2	91.1 ± 2.7	79.4 ± 3.2	0.61 ± 0.02
OPAL [23]	74.3 ± 3.4	64.2 ± 6.5	88.2 ± 4.4	78.8 ± 2.8	0.58 ± 0.04
OPAL Tran.	75.0 ± 2.2	64.9 ± 4.5	89.4 ± 3.3	79.2 ± 3.4	0.60 ± 0.05
HsO-VP Raw	75.7 ± 4.8	60.8 ± 2.8	83.3 ± 3.8	77.8 ± 3.3	0.55 ± 0.08
HsO-VP w/o BC	80.1 ± 2.3	70.7 ± 1.5	90.5 ± 1.2	81.6 ± 2.6	0.63 ± 0.03
HsO-VP	**82.8 ± 1.8**	**71.7 ± 2.5**	**94.9 ± 2.8**	**83.4 ± 2.9**	**0.65 ± 0.05**
HsO-VP Tran.	84.7 ± 1.4	73.5 ± 2.4	96.6 ± 2.2	85.6 ± 2.8	0.67 ± 0.04
Expert [34]	85.5 ± 2.6	75.0 ± 4.1	100.0 ± 0.0	86.4 ± 3.9	0.67 ± 0.02

B. Driving Performance

First, we implement baselines and HsO-VP, and test them at training (Town03) and new (Town05) driving scenarios (**Q1**). Results are recorded in Tab. I and Tab. II. Notably, planners achieve higher driving scores in the new scenario because Town03 is inherently more complex than Town05 [1].

Analyzing results in the tables, it becomes evident that offline RL algorithms (CQL [30] and IQL [31]) outperform IL algorithms (BC and MARWIL [48]) at both training and new scenarios, in line with our claims in the preceding texts. Furthermore, we observe that hierarchical offline RL algorithms (IRIS [40] and OPAL [23]) can indeed better address long-horizon planning tasks, achieving performance improvement compared to CQL and IQL. However, upon closer examination, it is evident that OPAL shows only limited improvement in driving scores compared to vanilla IQL. Specifically, there are only a 1.4% and a 0.9% improvement in training and new scenarios, respectively. This suggests that OPAL may suffer from posterior collapse [26, 27] and fails to extract highly effective skills, as depicted in Fig. 4(a).

In contrast, when we reformulate the generation process and introduce additional discrete skill variables, the trained HsO-VP planner achieves optimal results across metrics at both training and new scenarios. For the most significant driving score metric, it outperforms the strongest baseline IRIS by about 4.0% in absolute value at training scenarios and 6.4% at new scenarios. The significantly higher infraction score and normalized reward indicate that HsO-VP obtains higher driving efficiency and safety simultaneously. These results suggest that the hierarchical skills extracted by HsO-VP enable long-term reasoning for offline RL algorithms.



(a) OPAL Skill Extraction (b) Skill Extraction (c) Skill Semantics

Fig. 4: Visualization of learned skills. (a): T-SNE [49] visualization of skill embeddings learned by OPAL [23]. (b): T-SNE visualization of skill embeddings z (i.e. continuous skills) output by HsO-VP's sequence encoder, where the color labels are determined by y (i.e. discrete skills) obtained after the gumbel-softmax operation [29]. (c): Visualization of the skill sequences extracted from the corresponding areas in the left image.

C. Ablation Study

The key component of HsO-VP is its two-branch VAE structure. If a vanilla VAE is used instead, the algorithm just degenerates into OPAL [23], which has been examined in Sec. V-B to be significantly weaker than HsO-VP. In this section, we primarily conduct ablations on the training process of HsO-VP (Q2): First, the variant that directly uses extracted skills for policy initialization and tests without downstream training, is denoted as 'HsO-VP Raw'; Second, the variant that only trains the high-level policy using offline RL but does not finetune the low-level policy by BC, is called 'HsO-VP w/o BC'. The results are recorded in Tab. I and Tab. II.

The ablation results experiments show pronounced differences. We find that both discarding the finetuning of low-level policies and offline RL of high-level policies lead to performance losses across all metrics. The impact of not performing low-level finetuning is relatively small, indicating that the latent variables in the skill extraction phase are already well aligned with the execution of low-level actions. However, 'HsO-VP Raw' only obtains limited performance improvement compared to IL and is inferior to vanilla offline RL algorithms. Essentially, the skill extraction process does not consider the reward signals but simply reconstructs expert actions, albeit considering actions over multiple timesteps. Therefore, the skill-based offline RL process is crucial for the performance of HsO-VP.

D. Skill Analysis

In this section, we explore the properties of extracted skills from HsO-VP through further experiments (Q3).

Interpretability. In Fig. 4(b), we visualize the final output z from the two-branch sequence encoder using t-SNE [49]. We analyze the role of the discrete layer by using latent variables y obtained from the discrete branch as labels. As is in the figure, sequences with the same color are predominantly clustered together. This suggests that the gumbel-softmax operation [29] assigns different discrete skill options to different networks in the ensemble, thereby amplifying the differences between skill embeddings z. Hence, we observe that the extracted skills of HsO-VP are more distinguishable than OPAL [23] in Fig. 4(a). In Fig. 4(c), we visualize representative sequences from selected clusters. It can be seen that different colored blocks correspond to distinct discrete skill choices (from 0 to 4, right turn, stop, accelerate, left turn, and go straight, respectively). Furthermore, the clusters 0.a and 0.b in the same color (i.e. within the same skill option) stand for sharp and mild right turns, respectively, indicating flexible execution styles for actions in one discrete skill. These results demonstrate that HsO-VP captures both the discrete options and continuous variations of skills, providing evidence for the interpretability of skills.

Transferability [21]. In Sec. V-B, we have deployed HsO-VP at new driving scenarios and validated its strong performance. Here we study a different setting: *leverage the skill extraction module obtained from training scenarios to label skills at new (testing) scenarios, then use them instead of original training data for downstream training.* Essentially, we are investigating the transferability of learned skills, analyzing whether they can be directly used to facilitate planning in unseen driving tasks. The results are also recorded in Tab. II, highlighted in green and labeled as 'Tran.'. Notably, 'OPAL Tran.' only obtains 0.7% driving score improvement compared to OPAL, implying that the learned skills from training data do not align with data from new driving tasks. In contrast, 'HsO-VP Tran.' shows significant improvements across all metrics compared to 'HsO-VP'. These results prove that HsO-VP succeeds in learning skills that can be transferred to assist planning in new tasks.

VI. Conclusion and Outlooks

In this paper, we propose HsO-VP to boost offline RL for long-horizon vehicle planning. Specifically, we filter offline driving data and design a two-branch VAE to extract hierarchical skills that capture both discrete options and continuous variations, overcoming posterior collapse. Based on this, we train a high-level policy that outputs skills instead of per-step actions, which serve as temporal abstractions to enable long-term reasoning into the future. Comprehensive experimental results on CARLA prove that HsO-VP extracts interpretable driving skills while consistently outperforming strong baselines at both seen and unseen driving scenarios.

As a pioneering work to leverage skill-based offline RL for vehicle planning, we believe that HsO-VP will be a promising and inspiring framework to enable practical autonomous driving. There are quite a few exciting follow-up directions, including but not limited to learning driving skills from human-annotated skill data, a more refined generative process that can capture transitions between skills, extracting variable-length driving skills, etc.

REFERENCES

[1] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*, pp. 1–16, PMLR, 2017.

[2] Q. Sun, X. Huang, B. C. Williams, and H. Zhao, "Intersim: Interactive traffic simulation via explicit relation modeling," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 11416–11423, IEEE, 2022.

[3] H. Caesar, J. Kabzan, K. S. Tan, W. K. Fong, E. Wolff, A. Lang, L. Fletcher, O. Beijbom, and S. Omari, "nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles," *arXiv preprint arXiv:2106.11810*, 2021.

[4] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. R. Qi, Y. Zhou, *et al.*, "Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9710–9719, 2021.

[5] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635, JMLR Workshop and Conference Proceedings, 2011.

[6] A. Prakash, A. Behl, E. Ohn-Bar, K. Chitta, and A. Geiger, "Exploring data aggregation in policy learning for vision-based urban autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11763–11773, 2020.

[7] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[8] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International conference on machine learning*, pp. 2052–2062, PMLR, 2019.

[9] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims, "Morel: Model-based offline reinforcement learning," *Advances in neural information processing systems*, vol. 33, pp. 21810–21823, 2020.

[10] T. Shi, D. Chen, K. Chen, and Z. Li, "Offline reinforcement learning for autonomous driving with safety and exploration enhancement," *arXiv preprint arXiv:2110.07067*, 2021.

[11] C. Diehl, T. Sievernich, M. Krüger, F. Hoffmann, and T. Bertran, "Umbrella: Uncertainty-aware model-based offline reinforcement learning leveraging planning," *arXiv preprint arXiv:2111.11097*, 2021.

[12] J. Li, C. Tang, M. Tomizuka, and W. Zhan, "Hierarchical planning through goal-conditioned offline reinforcement learning," *arXiv preprint arXiv:2205.11790*, 2022.

[13] T. Zhou, L. Wang, R. Chen, W. Wang, and Y. Liu, "Accelerating reinforcement learning for autonomous driving using task-agnostic and ego-centric motion skills," *arXiv preprint arXiv:2209.12072*, 2022.

[14] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," *Advances in neural information processing systems*, vol. 30, 2017.

[15] E. Chane-Sane, C. Schmid, and I. Laptev, "Goal-conditioned reinforcement learning with imagined subgoals," in *International Conference on Machine Learning*, pp. 1430–1440, PMLR, 2021.

[16] E. Bronstein, M. Palatucci, D. Notz, B. White, A. Kuefler, Y. Lu, S. Paul, P. Nikdel, P. Mougin, H. Chen, *et al.*, "Hierarchical model-based imitation learning for planning in autonomous driving," *arXiv preprint arXiv:2210.09539*, 2022.

[17] J. Jing, E. C. Cropper, I. Hurwitz, and K. R. Weiss, "The construction of movement with behavior-specific and behavior-independent modules," *Journal of Neuroscience*, vol. 24, no. 28, pp. 6315–6325, 2004.

[18] L. Wang, Y. Hu, L. Sun, W. Zhan, M. Tomizuka, and C. Liu, "Transferable and adaptable driving behavior prediction," *arXiv preprint arXiv:2202.05140*, 2022.

[19] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[20] C. Zhang, R. Guo, W. Zeng, Y. Xiong, B. Dai, R. Hu, M. Ren, and R. Urtasun, "Rethinking closed-loop training for autonomous driving," in *European Conference on Computer Vision*, pp. 264–282, Springer, 2022.

[21] K. Pertsch, Y. Lee, and J. Lim, "Accelerating reinforcement learning with learned skill priors," in *Conference on robot learning*, pp. 188–204, PMLR, 2021.

[22] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, "Learning latent plans from play," in *Conference on robot learning*, pp. 1113–1132, PMLR, 2020.

[23] A. Ajay, A. Kumar, P. Agrawal, S. Levine, and O. Nachum, "Opal: Offline primitive discovery for accelerating offline reinforcement learning," in *International Conference on Learning Representations*, 2021.

[24] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[25] C. Zhang, J. Bütepage, H. Kjellström, and S. Mandt, "Advances in variational inference," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 2008–2026, 2018.

[26] Y. Wu, Y. Burda, R. Salakhutdinov, and R. Grosse, "On the quantitative analysis of decoder-based generative models," in *International Conference on Learning Representations*, 2017.

[27] C. P. Burgess, I. Higgins, A. Pal, L. Matthey, N. Watters, G. Desjardins, and A. Lerchner, "Understanding disentangling in $beta$-vae," *arXiv preprint arXiv:1804.03599*,

2018.

[28] A. Vahdat and J. Kautz, "Nvae: A deep hierarchical variational autoencoder," *Advances in neural information processing systems*, vol. 33, pp. 19667–19679, 2020.

[29] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[30] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1179–1191, 2020.

[31] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," *arXiv preprint arXiv:2110.06169*, 2021.

[32] M. Bansal, A. Krizhevsky, and A. Ogale, "Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst," *arXiv preprint arXiv:1812.03079*, 2018.

[33] S. Starke, H. Zhang, T. Komura, and J. Saito, "Neural state machine for character-scene interactions.," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 209–1, 2019.

[34] Z. Zhang, A. Liniger, D. Dai, F. Yu, and L. Van Gool, "End-to-end urban driving by imitating a reinforcement learning coach," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15222–15232, 2021.

[35] A. Hu, G. Corrado, N. Griffiths, Z. Murez, C. Gurau, H. Yeo, A. Kendall, R. Cipolla, and J. Shotton, "Model-based imitation learning for urban driving," in *Advances in Neural Information Processing Systems*, 2022.

[36] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for bertha—a local, continuous method," in *2014 IEEE intelligent vehicles symposium proceedings*, pp. 450–457, IEEE, 2014.

[37] J. Chen, W. Zhan, and M. Tomizuka, "Autonomous driving motion planning with constrained iterative lqr," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 2, pp. 244–254, 2019.

[38] S. Fujimoto and S. S. Gu, "A minimalist approach to offline reinforcement learning," *Advances in neural information processing systems*, vol. 34, pp. 20132–20145, 2021.

[39] Y. Wu, S. Zhai, N. Srivastava, J. M. Susskind, J. Zhang, R. Salakhutdinov, and H. Goh, "Uncertainty weighted actor-critic for offline reinforcement learning," in *International Conference on Machine Learning*, pp. 11319–11328, PMLR, 2021.

[40] A. Mandlekar, F. Ramos, B. Boots, S. Savarese, L. Fei-Fei, A. Garg, and D. Fox, "Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4414–4420, IEEE, 2020.

[41] T. Nam, S.-H. Sun, K. Pertsch, S. J. Hwang, and J. J. Lim, "Skill-based meta-reinforcement learning," in *International Conference on Learning Representations*, 2022.

[42] E. Rosete-Beas, O. Mees, G. Kalweit, J. Boedecker, and W. Burgard, "Latent plans for task-agnostic offline reinforcement learning," in *6th Annual Conference on Robot Learning*, 2022.

[43] N. Deo, A. Rangesh, and M. M. Trivedi, "How would surround vehicles move? a unified framework for maneuver classification and motion prediction," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 129–140, 2018.

[44] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[45] M. Ahmed, R. Seraj, and S. M. S. Islam, "The k-means algorithm: A comprehensive survey and performance evaluation," *Electronics*, vol. 9, no. 8, p. 1295, 2020.

[46] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *International conference on learning representations*, 2017.

[47] C. team, "Carla autonomous driving leaderboard," 2020.

[48] Q. Wang, J. Xiong, L. Han, H. Liu, T. Zhang, *et al.*, "Exponentially weighted imitation learning for batched historical data," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[49] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.