Lightweight Strategies for Decision-Making of Autonomous Vehicles in Lane Change Scenarios Based on Deep Reinforcement Learning

Guofa Li, *Senior Member, IEEE*, Jun Yan, Yifan Qiu, Qingkun Li, *Member, IEEE*, Jie Li, Shengbo Eben Li, *Senior Member, IEEE*, and Paul Green

***Abstract*— High-performance vision-based decision-making networks are often limited by hardware capabilities in practical applications. To address this challenge, this study proposes lightweight optimization strategies for decision-making models from the aspects of parameter size, training memory usage, and inference speed. Specifically, an innovative solution is proposed to achieve lightweight parameters. The Video Swin Transformer is employed to simultaneously extract temporal and spatial features, with the network trained using a Prioritized Replay Deep Q-Network (PRDQN) that incorporates risk assessment. To further reduce training memory usage, the Q-target network in PRDQN is removed, and the mellowmax operator is integrated to enhance the training process, resulting in the PRDeepMellow Swin Transformer. After analyzing the inference speed problems encountered by the algorithm in practical applications, the vanilla self-attention is replaced by a linear self-attention based on double softmax, namely Double Softmax Linear Video Swin Transformer (DSLVS Transformer) which improves the inference speed for long sequences. The proposed methods were evaluated across three high-speed lane change scenarios (a static scenario, a dynamic scenario, and a randomly changing scenario). Experimental results demonstrate that the proposed methods can still maintain excellent decision performance after the corresponding lightweight optimizations.**

***Index Terms*— Driving safety, autonomous vehicles, lane change, decision making, reinforcement learning, light-weight.**

I. Introduction

DECISION-MAKING is regarded as a critical aspect for the application of autonomous driving technologies [1], [2], [3], [4]. However, existing decision-making models based on visual image inputs often exhibit an overreliance on the number of parameters within the neural network [5], [6]. While current research primarily focuses on improving the accuracy and robustness of decision-making models [7], [8], [9], [10], [11], [12], challenges such as deployment difficulties, long inference times, and high training costs remain largely overlooked. Lightweight model deployment and optimization methods that can be deployed in practical applications are urgently needed by the industry [13]. Current lightweight approaches generally focus on three main aspects: data, model, and framework.

A. Lightweight Approaches on Data

Lightweight approaches for data aim to eliminate redundant information in input data or feature maps. Henning et al. [14] introduced a multi-layer attention map (MLAM) that includes an input data filtering mechanism to process only relevant data, effectively reducing redundancy in environmental perception feature extraction. Hua et al. [15] proposed a method called channel gating. This method optimizes CNN inference by leveraging input-specific features and establishing a feature map filtering mechanism. By dynamically skipping the propagation of weights in unimportant regions, channel gating significantly reduces the computational overhead and computational burden.

Convolutional neural networks (CNNs) remain the dominant architecture for perception models in the industry. Substantial research has been conducted to enhance their lightweight efficiency. Meanwhile, Vision Transformers (ViT) have emerged as powerful tools for feature extraction through attention mechanisms. However, the lightweight potential of pure attention mechanisms remains theoretical, with limited practical application in autonomous driving. For instance, Rao et al. [16] developed a lightweight data encoding method using ViT with dynamic token scarification. This approach modulates the number of tokens passed to subsequent layers, thereby accelerating inference.

B. Lightweight Approaches on Models

Lightweight approaches for models in deep learning commonly include pruning, quantization, knowledge distillation, low-rank approximation, lightweight structural design, automated machine learning (AutoML), and neural architecture

Received 27 August 2024; revised 3 December 2024; accepted 21 January 2025. Date of publication 4 February 2025; date of current version 5 May 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 52272421 and in part by the State Key Laboratory of Intelligent Green Vehicle and Mobility under Project KFZ2409. The Associate Editor for this article was C. Wei. *(Corresponding author: Jie Li.)*

Guofa Li, Jun Yan, and Jie Li are with the College of Mechanical and Vehicle Engineering, Chongqing University, Chongqing 400044, China (e-mail: liguofa@cqu.edu.cn; yanjun4611@gmail.com; jieli14@tsinghua.org.cn).

Yifan Qiu is with the Institute of Human Factors and Ergonomics, College of Mechatronics and Control Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: rye1222@qq.com).

Qingkun Li is with Beijing Key Laboratory of Human-Computer Interaction, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China (e-mail: liqingkun@iscas.ac.cn).

Shengbo Eben Li is with the School of Vehicle and Mobility, Tsinghua University, Beijing 100084, China (e-mail: lishbo@tsinghua.edu.cn).

Paul Green is with the University of Michigan Transportation Research Institute (UMTRI) and the Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: pagreen@umich.edu).

Digital Object Identifier 10.1109/TITS.2025.3534797

search (NAS) [17]. One of the most iconic lightweight design approaches for convolutional neural network structures is depthwise separable convolution [18], which allows convolution computations to be reduced to just one-ninth of those required for standard convolutions. Similarly, various lightweight architectural designs have been developed for attention-based approaches within Transformer models.

Hechen et al. [19] introduced Light Self-Limited Attention (LSLA), a lightweight self-attention mechanism that simplifies computation by replacing the key and value components with the input itself. LSLA incorporates positional information and limits attention weights using external biases, resulting in the hierarchical ViT-LSLA. Venkataramanan et al. [20] proposed a Skip-Attention mechanism to reuse self-attention computations from earlier layers, significantly accelerating inference without compromising performance.

In pruning, one major challenge is the dependency between algorithm implementation and network architecture, which complicates generalizability. Fang et al. [21] addressed this issue with DepGraph, an architecture-agnostic structured pruning method applicable to CNNs, Transformers, and RNNs. DepGraph accurately removes redundant parameters by analyzing structural dependencies, improving model efficiency across diverse architectures.

C. Lightweight Approaches on Frameworks

Framework-level lightweight research addresses challenges such as hardware inefficiency and deployment constraints. Conventional pruning methods often struggle with hardware incompatibility and computational overhead, especially in autonomous driving applications. To overcome these limitations, Li et al. [22] proposed the Dynamic Width Variable Network, a pruning method that dynamically adjusts filter activations based on input complexity. This approach enables efficient parameter slicing for hardware-friendly implementation.

In terms of operator architecture, Baidu's PaddlePaddle collaborated with the Apollo Open Platform to optimize GPU utilization for autonomous driving scenarios [23]. By incorporating acceleration libraries such as TensorRT and OpenVINO, this partnership achieved significant improvements in inference speed and computational efficiency for tasks like point cloud voxelization and 3D Non-Maximum Suppression (NMS).

Additionally, lightweight solutions have been explored for decision-making in deep reinforcement learning. Kim et al. [24] proposed the use of a single value estimation network to replace multiple networks in reinforcement learning training, which helps reduce memory usage. Although promising, this method remains theoretical and lacks application in autonomous driving. Huawei's lightweight positioning algorithm RoadMap [25] offers another perspective, compressing high-precision maps into semantic representations with minimal storage requirements. This approach achieves real-time localization with an average map size of just 36 kb/km.

A review of current lightweight research shows notable progress in three domains: data, models, and frameworks. Efforts in the data domain focus on reducing input redundancy and computational demands, while model-level research targets improved computational efficiency and faster inference. Framework-related studies aim to minimize deployment overhead and enhance hardware compatibility. However, lightweight solutions specifically tailored for autonomous driving decision-making remain underexplored. Developing effective lightweight models for this purpose is a critical and pressing challenge with far-reaching implications for advancing autonomous driving technology.

D. Contributions

Given the requirements for practical deployment of decision-making model, three key aspects must be prioritized in the lightweight design: the size of the model parameters, the training memory usage, and the speed of inference. Firstly, to address the task of reducing the number of model parameters, a vision-based lane change autonomous driving decision-making model is proposed. This model integrates the Video Swin Transformer with risk-assessment based PRDQN algorithm. Then, to reduce memory usage during training, the PRDeepMellow Swin Transformer is introduced, which improved the reinforcement learning training process through the use of the mellowmax operator. To improve inference speed, the DSLVS Transformer is presented, which replaces the vanilla self-attention with linear self-attention. Lastly, three lane change decision-making scenarios in the CARLA are established to evaluate the performance of the proposed algorithms. The main contributions of this study include:

(1) An innovative lightweight lane change decision model is proposed that uses the Video Swin Transformer to simultaneously extract temporal and spatial features, and integrates a risk assessment function to address the deployment challenges posed by excessive model parameters.
(2) By integrating the mellowmax operator and eliminating the Q-target network, the PRDeepMellow Swin Transformer is innovatively developed. This effectively reduces the training memory requirements.
(3) An innovative approach replaces the vanilla self-attention of Video Swin Transformer with a double-softmax-based linear self-attention, namely DSLVS Transformer. This improves the inference speed for long sequence problems.

II. Related Work

A. Reinforcement Learning

1) Q-Learning: Q-learning is a method where an agent continuously interacts with the environment to optimize the action-value function $Q(s, a)$:

$$Q(s,a) = E\left[r(s,a) + \gamma E\left[Q\left(s', a'\right)\right]\right] \tag{1}$$

where s, a are the current state and action, s', a' are the next state and action, $r(s, a)$ is the reward from taking action a at state s. Over the years, Q-learning has proven to be a highly effective reinforcement learning algorithm. However, it can only handle problems with small state and action spaces, which limits its application to complex problems.

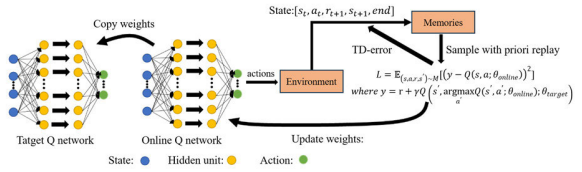


Fig. 1. The framework of PRDQN.

2) Deep Q-Network (DQN): To enable the application of the Q-learning algorithm to complex problems, the successful combination of deep convolutional neural networks with Q-learning was achieved by Mnih et al. [26], effectively avoiding the curse of dimensionality in complex spaces. Additionally, the concepts of experience replay and target networks were introduced by the DQN algorithm. Experience replay enables offline learning by breaking the correlation between consecutive samples, thereby increasing the efficiency of sample utilization. While training the Q-network, the DQN algorithm also trains a Q-target network. These two networks are relatively independent but share the same structure. The Q-network is used to calculate the Q-values of different actions under the current policy in real-time, while the Q-target network provides stable Q-value estimations to reduce instability during the training process.

3) Prioritized Replay Deep Q-Network (PRDQN): The experience replay mechanism in the DQN algorithm uses uniform sampling, which leads to the under-sampling of rare but valuable samples and thereby reduces training efficiency. To address this issue, the PRDQN algorithm was proposed by Hessel et al. [27]. Samples with larger TD errors are prioritized by PRDQN, making them more likely to be sampled and learned. The definition of priority is described as:

$$P(i) = \frac{p_i^\gamma}{\sum_k p_k^\gamma} \tag{2}$$

where p represents the TD error and γ is a constant.

The calculation of the loss function for updating the Q-network in the PRDQN algorithm is described as:

$$\begin{aligned} L &= \mathbb{E}_{(s,a,r,s')\sim M}\left[(y - Q(s,a;\theta_{online}))^2\right] \\ y &= \mathrm{r} + \gamma Q\left(s', Q\left(s', a'; \theta_{online}\right); \theta_{target}\right) \end{aligned} \tag{3}$$

where (s, a, r, s') is a trajectory sampled in memory M, θ_{online} and θ_{target} are the weights of Q-network and Q-target network.

The complete computational flow of PRDQN is depicted in Fig. 1.

B. Mellowmax Operator

Mellowmax is an improved operator designed to overcome the decision imbalance problem of the softmax operator under extreme conditions [24]:

$$mm_\omega = \frac{log\left(\frac{1}{n}\sum_{i=1}^n e^{\omega x_i}\right)}{\omega} \tag{4}$$

As illustrated by the preceding Eq. (4), mellowmax controls the operation result by performing a scaling transformation on the input feature space using a hyperparameter ω (where $\omega > 0$). When ω takes extreme values, the mellowmax operator can be expressed as:

(1) The mellowmax operator approaches the max operator when ω tends towards positive infinity:

$$\begin{aligned} \lim_{\omega\to+\infty} mm_\omega &= \lim_{\omega\to+\infty} \frac{log\left(\frac{1}{n}\sum_{i=1}^n e^{\omega x_i}\right)}{\omega} \\ &= \lim_{\omega\to+\infty} \frac{\log\left(\frac{1}{n}e^{\omega\max(x_i)}\sum_{i=1}^n e^{\omega(x_i-\max(x_i))}\right)}{\omega} \\ &= \lim_{\omega\to+\infty} \frac{\log\left(e^{\omega\max(x_i)}\right) - \log(n) + \log\left(\sum_{i=1}^n e^{\omega(x_i-\max(x_i))}\right)}{\omega} \\ &= \max(x_i) + \lim_{\omega\to+\infty} \frac{-\log(n) + \log\left(\sum_{i=1}^n e^{\omega(x_i-\max(x_i))}\right)}{\omega} \end{aligned}$$

where $x_i - max\,(x_i) < 0$, $e^{\omega(x_i-\max(x_i))} \to 0$

$$\lim_{\omega\to+\infty} mm_\omega = max\,(x_i) \tag{5}$$

(2) The mellowmax operator approaches the mean operator when ω tends towards zero:

$$\begin{aligned} \lim_{\omega\to\infty} mm_\omega &= \lim_{\omega\to\infty} \frac{\log\left(\frac{1}{n}\sum_{i=1}^n e^{\omega x_i}\right)}{\omega} \\ &\xrightarrow{\text{L Hospital Theory}} \lim_{\omega\to\infty} \frac{\frac{1}{n}\sum_{i=1}^n x_i e^{\omega x_i}}{\frac{1}{n}\sum_{i=1}^n e^{\omega x_i}} \\ &= \frac{1}{n}\sum_{i=1}^n x_i \end{aligned} \tag{6}$$

Given that different values of ω can result in varying performance of the mellowmax operator, it is essential to study and analysis of ω in practical applications.

C. Attention Mechanism

1) Multi-Head Self-Attention: The Transformer is a sequence feature extraction network based on attention, featuring an encoder-decoder structure. The correlation between two input tokens is calculated, enabling each token to carry information from other tokens. This approach effectively addresses the long sequence dependency issue inherent in recurrent neural networks and is highly conducive to parallel computing.

Multi-Head Self-Attention and Masked Multi-Head Self-Attention are key components in the encoder and decoder in the Transformer structure. They serve as crucial modules for feature extraction. The attention in the decoder is different from that in the encoder. Tokens must be masked to prevent interference from future, unobserved output tokens when computing the current token. The attention module is based on a scaled dot-product self-attention. Query vectors Q, key vectors K, and value vectors V are derived from the input mappings. The computational principle is described as:

$$\begin{aligned} Attention(Q, K, V) &= Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \\ Q = W_Q X, K &= W_K X, V = W_V X \end{aligned} \tag{7}$$

where d_k is the dimension of K.

The basic principle of the multi-head self-attention is that the input is processed by dividing it into multiple heads. Different weight matrices are used by each head to extract

features from different perspectives. This process is described as:

$$MultiAttention\,(Q,K,V)=Concat\,(head_1,\ldots,head_h)\,W_o$$
$$head_i = Attention\left(W_Q^i X_i, W_K^i X_i, W_V^i X_i\right) \tag{8}$$

2) Linear Self-Attention: Linear self-attention is introduced to mitigate the issue of vanilla self-attention's rapidly increasing computational complexity with longer input sequences. Algorithm 1 outlines the primary steps and computational complexity of self-attention, excluding the softmax operation and scale normalization for simplicity. The $\bigotimes$ symbol represents the matrix multiplication, n denotes the length of the sequence, d denotes the number of feature channels extracted by each self-attention head, h denotes the number of self-attention heads, and hd denotes the number of feature channels extracted by the self-attention module, where $X, Q, K, V \in R^{n\times hd}$.

The analysis shows that the computational complexity of self-attention is $O\left(n^2\right)$, meaning the cost grows quadratically with sequence length. As detailed in Algorithm 1, the $O\left(n^2\right)$ computation is primarily generated during the matrix multiplication associated with the self-attention. Examining the self-attention formula reveals that this multiplication is incorporated into the softmax operator [28]. By removing the softmax operator, self-attention reduces to the simple concatenation of the matrix product QK^TV. Based on the principle of multiplicative union, if K^TV is computed first, step 2 in Algorithm 1 can be reformulated as shown in Algorithm 2.

The computational complexity is reduced from $O\left(n^2\right)$ to $O\,(n)$, making the removal of the softmax operator critical for establishing linear self-attention. By expanding and generalizing Eq. (7), the following definition can be obtained:

$$\begin{aligned} Attention\,(Q,K,V)_i &= Softmax\left(\frac{Q_i K^T}{\sqrt{d_k}}\right)V \\ &\xrightarrow{Neglect\sqrt{d_k}\ and\ Expand\ Softmax} \frac{\sum_{j=1}^{n} e^{q_i k_j^T}\cdot v_j}{\sum_{j=1}^{n} e^{q_i k_j^T}} \\ &\xrightarrow{\text{Generalize}} \frac{\sum_{j=1}^{n} sim\left(q_i,k_j\right)\cdot v_j}{sim\left(q_i,k_j\right)} \end{aligned} \tag{9}$$

where $sim(\cdot)$ is a generalized function. To maintain a distribution similar to self-attention, it is necessary to impose the constraint $sim(q_i,k_j)\geq 0$.

However, the vanilla dot product does not satisfy this constraint. To ensure non-negativity, it is essential to apply an activation function to both Q and K, as demonstrated below:

$$sim\left(q_i,k_j\right)=\phi\,(q_i)^T\varphi\left(k_j\right) \tag{10}$$

where $\phi\,(\cdot)$ and $\varphi(\cdot)$ are activation functions with non-negative ranges. When $\phi = \varphi$, ϕ can be regarded as a kernel function [29], with $sim(q_i,k_j)$ representing the inner product of these kernel functions. Different forms of $\phi\,(\cdot)$ and $\varphi(\cdot)$ result in various linear self-attention mechanisms, satisfying the condition $e^{q_i k_j^T}\approx\phi\,(q_i)^T\varphi(k_j)$ [3 0]. This decomposition of the product QK allows for a reduction in computational complexity to a linear scale.

D. DRL-Based Baseline Model for Vision Decision-Making

The work in Li et al. [31] investigates a high-performance application as a baseline model for autonomous driving, with a focus on the task of lane change decisions at high speeds. The feasibility of the proposed solution is validated in an autonomous driving simulation environment. In the decision layer, PRDQN is utilized, while the perception layer is constructed using depthwise separable convolutional neural networks (DSCNN) for spatial feature extraction and a Transformer for temporal feature extraction. This combination forms the DSCNN Transformer lane change decision-making network based on spatiotemporal hybrid convolutions. See Fig. 2. The DSCNN Transformer architecture is effectively used to complete lane change driving decisions. However, the challenge is to adapt this large-scale baseline model to the practical requirements of complex driving environments and to facilitate its seamless deployment on vehicle-embedded

Algorithm 1 Computational Complexity of Vanilla Self-Attention

Step 1: Project the input X into Q, K, V:
Dimensional Transformation: $n\times hd\bigotimes hd\times n$, Computation: $3n\,(hd)^2$
Step 2: Computation of one Head in Self-Attention:
2-1: Computation of Attention Weight Matrices for Each h Heads $attn = QK^T$:
Dimensional Transformation: $n\times d\bigotimes d\times n$, Computation: n^2d
2-2: Applying Weights to V in Each h Heads:
Dimensional Transformation: $n\times n\bigotimes n\times d$, Computation: n^2d
Total Computation of the Module: $2n^2hd$
Step 3: Fuse Multi-Head Attention through MLP:
Dimensional Transformation: $n\times hd\bigotimes hd\times hd$, Computation: $n\,(hd)^2$
Total Computation of the Self-Attention Module: $4n\,(hd)^2+2n^2hd$

Algorithm 2 Computational Complexity of Linear Self-Attention

Step 2: Computation of one Head in Self-Attention:
2-1: Computation of Attention Weight Matrices for Each h Heads $attn = K^TV$:
Dimensional Transformation: $d\times n\bigotimes n\times d$, Computation: nd^2
2-2: Applying Weights to Q in Each h Heads:
Dimensional Transformation: $n\times d\bigotimes d\times d$, Computation: nd^2
Total Computation of the Module: $2nhd^2$
Total Computation of the Improved Self-Attention Module: $4n\,(hd)^2+2nhd^2$

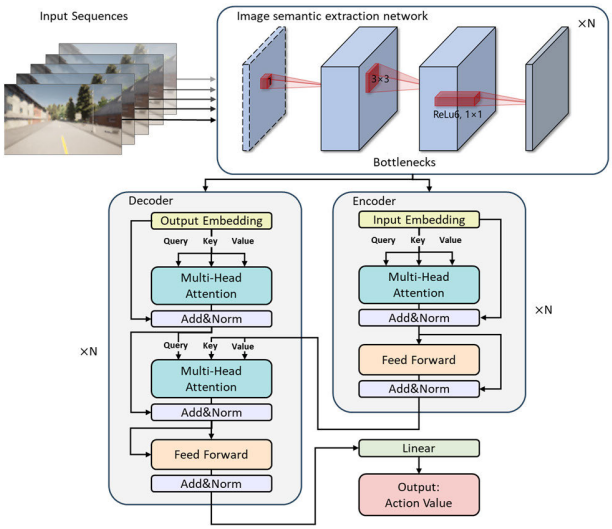


Fig. 2. DSCNN transformer decision network by spatiotemporal hybrid convolutions.

systems. Therefore, a lightweight enhancement approach tailored to the baseline model needs to be explored. This study provides a critical foundation for future models that can be effectively deployed in complex environments and to advance high-performance decision-making methodologies.

III. Proposed Lightweight Method

In this section, three lightweight optimizations are innovatively proposed including optimization of model parameter size, efficient use of training memory, and enhancement of inference speed. The corresponding details are introduced as follows.

A. Small Parameter Size: Risk-Aware Spatial-Temporal Parallel Feature Extraction

To enhance the robustness and safety of vehicles during lane change decisions, the risk assessment has been integrated into the PRDQN algorithm for targeted neural network training. In order to minimize the number of model parameters, the Video Swin Transformer has been employed to concurrently extract temporal and spatial features from the input frames. The specific details of proposed model are shown in Fig. 3. Detailed descriptions of this architecture are provided in the following sub-sections.

1) Details of the PRDQN Algorithm Settings:

a) State space and action space: Considering the information obtainable by vehicle sensors in real driving scenarios, the states in RRDQN are defined using video frame inputs. The images captured in the foreground are obtained by a camera positioned directly above the vehicle's roof at a height of 1.65 meters (z =1.65m). The camera operates at a sampling frequency of 50 Hz and has a field of view (FOV) of 110 degrees. The images have a resolution of 144 pixels in height and 256 pixels in width.

The video frame inputs are composed of five images collected up to the current moment. The state space for the visual modality is defined as:

$$s = \left[I_{t-4}, I_{t-3}, I_{t-2}, I_{t-1}, I_t\right] \tag{11}$$

To prevent the DRL agent from adopting overly conservative behaviors (e.g., taking an excessive number of brake actions), the designed autonomous vehicle is responsible only for throttle and steering actions. Braking actions are expected to be controlled by a human driver or other braking programs. This approach ensures that the autonomous vehicle can effectively avoid collisions and safely complete driving tasks without relying on learned braking strategies. The action space at time t is defined as:

$$a_t \in \{LTL_t, LTS_t, S_t, RTS_t, RTL_t\} \tag{12}$$

where LTL_t and RTL_t represent significant steering for left and right turns, assigned values of ± 0.5 ($+$ for left and $-$ for right). LTS_t and RTL_t denote mild steering for left and right turns, with values of ± 0.1. The S_t represents the agent maintaining straight-line driving without any steering operation. During all maneuvers, the throttle value is consistently set to 0.5.

Certain DRL algorithms, such as PRDQN, operate solely within discrete action spaces. As a result, they often fail to ensure comfortable driving strategies and frequently generate rough trajectories. To address this, actions need to be smoothed by taking into account both the actions from the previous time step and those calculated in the current one. An exponential moving average strategy is employed for the generation of the actual actions:

$$a_t^* = a_{t-1} + \gamma\left(a_t - a_{t-1}\right) \tag{13}$$

where a_t^* is the actual action, γ is the hyperparameter for action smoothing, a_{t-1} is the action taken at the previous time step, and a_t is the theoretical action calculated at the current time step.

b) Risk evaluation: The method presented in this study employs a risk-based approach to determine the optimal lane change decision. This approach entails the calculation of the risk associated with potential obstacles in the driving environment. In the initial stage of the process, the risk levels are classified into three categories with a numerical value assigned to each:

$$\begin{aligned}\varepsilon \in \Omega &= \{Dangerous, Attentive, Safe\} \\ &= \{D = 2, A = 1, S = 0\}\end{aligned} \tag{14}$$

The risk level is defined as a measure of the likelihood of a collision between vehicles and is closely tied to their relative distance. To estimate collision probability, an uncertainty-based analysis method is applied. A Gaussian function is used as an efficient and effective means of converting distance into a corresponding risk value:

$$\begin{aligned}P\left(d \mid \varepsilon = D\right) &= \begin{cases} 1, & if\, d < d_D \\ e^{-\frac{\nabla d_D^2}{2\sigma^2}}, & otherwise \end{cases} \\ P\left(d \mid \varepsilon = A\right) &= e^{-\frac{\nabla d_A^2}{2\sigma^2}} \\ P\left(d \mid \varepsilon = S\right) &= \begin{cases} e^{-\frac{\nabla d_D^2}{2\sigma^2}}, & otherwise \\ 1, & if\, d > d_S \end{cases} \\ \nabla d_i &= \left|d - d_i\right|, \quad i \in \{D, A, S\}\end{aligned} \tag{15}$$

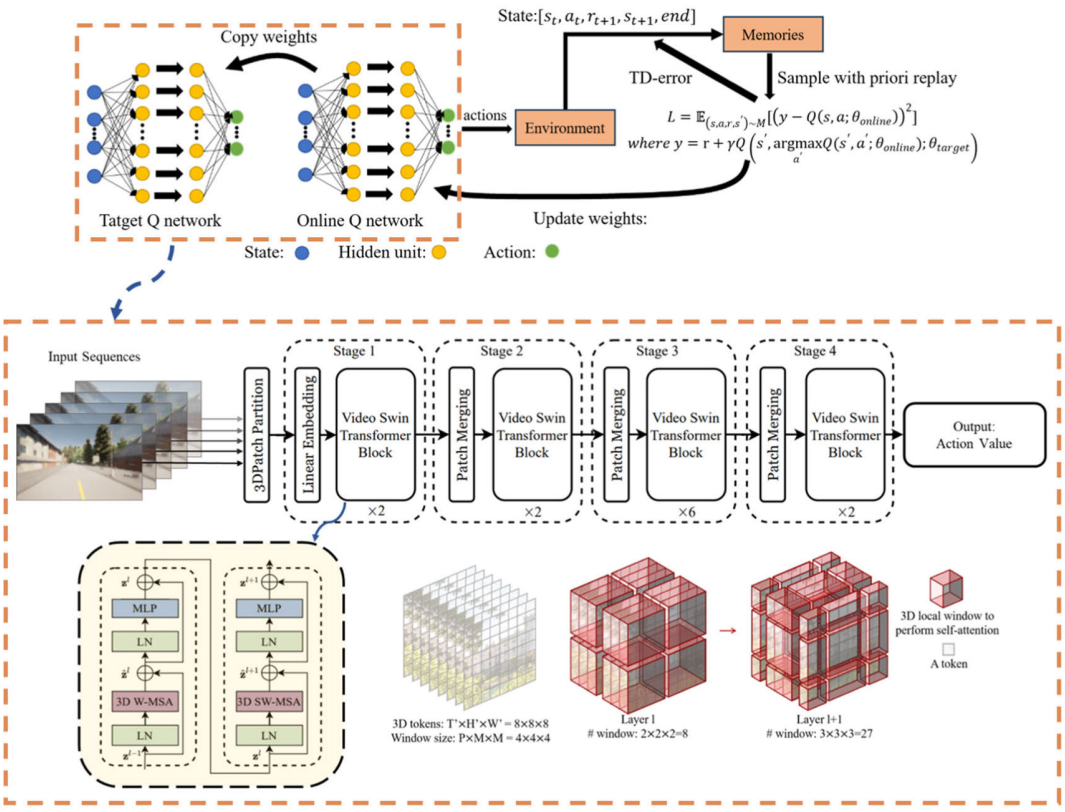


Fig. 3. Overall framework of the proposed model.

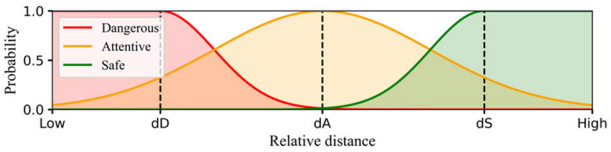


Fig. 4. The likelihood probability distribution for driving risk.

where d is the relative distance between the ego vehicle and the other vehicle, d_D, d_A, and d_S are the risk distance thresholds for different risk levels. σ denotes the uncertainty of the risk distance, which is generally associated with the shape of the aforementioned likelihood probability distribution. It is typically treated as a constant hyperparameter. Fig. 4 depicts the likelihood probability distribution, which illustrates that the risk increases as the relative distance decreases.

In order to refine the risk levels, it is necessary to make them continuous. In accordance with Bayes' theorem, the posterior probability of the risk level at a specified risk distance is calculated as follows:

$$P(\varepsilon|d) = \frac{P(d \mid \varepsilon) \cdot P(\varepsilon)}{\sum_{\varepsilon \in \Omega} P(\varepsilon) \cdot P(d \mid \varepsilon)} \tag{16}$$

where $P(\varepsilon)$ is the prior probability of the risk level. It is assumed that the prior distribution is uniform, and the sum of the probabilities for each risk level is equal to one, i.e., $\sum_{\tau \in \Omega} P(\tau) = 1$. Accordingly, the selection of the prior distribution has a minimal effect on the continuity of the results.

Subsequently, the probabilities of each vehicle falling into one of three predefined driving risk levels are calculated based on their specific relative distances. By computing the expectation, the continuous risk between the other vehicles and the ego vehicle can be obtained:

$$\varepsilon_d = \mathbb{E}(\varepsilon) = \sum_{\varepsilon \in \Omega} \varepsilon \cdot P(\varepsilon \mid d) \tag{17}$$

To obviate ambiguity in the symbolic representation of risk levels, the continuous risk ε_d is subsequently represented uniformly as the discrete risk level symbol ε, while the risk itself remains continuous.

c) Reward function: Developing an effective reward function is essential for promote proper lane change behavior in autonomous vehicles. The reward function is designed with consideration of the following three aspects:

(1) Driving Risk: The training target of PRDQN is to find a strategy that minimizes risk. When a negative sign is added, this part of the reward function can be expressed in the following form:

$$r_{risk} = \max_s \varepsilon - \varepsilon_t \tag{18}$$

where $\max_s \varepsilon$ is the maximum risk level under the current state.

(2) Task Completion: If the vehicle avoids collisions and remains within the lane boundaries during the designated time,

it will be rewarded:

$$r_{exist} = \begin{cases} 0.1, & if\ exist \\ -1, & otherwise \end{cases} \tag{19}$$

(3) Distance to Lane Boundary: The distance between the ego vehicle and the lane boundary should be calculated and represented as a soft reward using a Gaussian function. This approach assists the ego vehicle in keeping a reasonable distance to lane boundaries, thereby alleviating the issue of sparse rewards.

$$r_{invasion} = -e^{-\frac{(la_{lane}-la_{hv})^2}{2\sigma^2}} \tag{20}$$

where la_{lane} is the lane line location, la_{hv} is the horizontal position of the ego vehicle, and σ represents the uncertainty.

In conclusion, the reward function utilized in the experiments is represented as follows:

$$r = \lambda_1 \cdot r_{risk} + \lambda_2 \cdot r_{exist} + \lambda_3 \cdot r_{invasion}, \lambda_i \in [0, 1] \tag{21}$$

where λ_i is the weight of each reward function component. In this study, favorable results are achieved when $\lambda_1 = 1$, $\lambda_2 = 0.4$, $\lambda_3 = 0.3$.

2) Lightweight Lane Change Decision-Making Network: The DSCNN Transformer is a method of sequential serial extraction of spatial and temporal features. It is capable of efficiently extracting the features of input video frames and has been shown to perform well in the context of lane change decision-making tasks. However, the combination of two neural networks creates a model with a significant number of parameters, posing a challenge for practical application. To address this, this study explores using a single neural network for parallel extraction of temporal and spatial features, aiming to reduce the model's parameter count. In order to maintain performance, the recently successful Transformer has been selected for analysis in this study [32], [33].

Window Multi-head Self-Attention (W-MSA) offers a solution to the issue of exponential growth in the computational complexity of ViT with image size. Compared to Multi-head Self-Attention, the lightweight improvement provided by the W-MSA is described as:

$$\begin{aligned} \Omega(MSA) &= 4hwC^2 + 2(hw)^2C \\ \Omega(W-MSA) &= 4hwC^2 + 2M^2hwC \end{aligned} \tag{22}$$

where $\Omega(MSA)$ is the parameter size of the Multi-head Self-Attention, $\Omega(W-MSA)$ is the parameter size of the W-MSA, M is the window size, and h, w, and C are the height, width, and the number of channels in feature map.

In order to consider the interaction between windows, shift processing is carried out on the basis of the W-MSA to form the Shifted Window Multi-Head Self-Attention (SW-MSA). Building on this foundation, the local attention calculation is extended from the spatial domain to the spatial-temporal domain, resulting in the development of Video Swin Transformer. This neural network enables parallel extraction of temporal and spatial features.

The video frame input in 3D Pathc Partition is divided into 56 3D tokens, with each token comprising a 96-dimensional feature. The Video Swin Transformer employs a parallel approach to extract temporal and spatial features from the inputs. Based on these extracted features, the model assesses the degree of change in the level of risk and predicts the final action value. In order to represent its risk awareness capability, the trained neural network is named as Video Swin Transformer (RA).

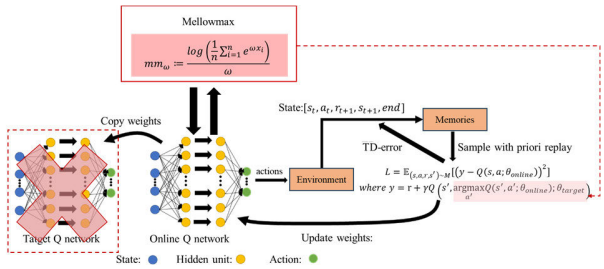


Fig. 5. The framework for PRDQN training without a Q-target network based on mellowmax.

B. Low Memory Usage: Improved Training Method to Mitigate Value Overestimation

In addition to the storage space necessary for model deployment, reinforcement learning also requires the use of memory to maintain an interactive simulation environment for training. If memory usage is not effectively minimized, the frequency of interactions between the algorithm and the environment will be adversely affected. Additionally, the model training speed may decrease, and the training process could even be halted due to insufficient memory resources.

The PRDQN algorithm primarily employs memory to maintain an experience replay buffer and two Q-networks. In this study, the Q-target network is eliminated by bring in mellowmax to improve the iterative process of the PRDQN algorithm, consequently reducing memory consumption during training. The improved algorithm is designated Prioritized Replay DeepMellow (PRDeepMellow). Fig. 5 depicts the training process of the PRDeepMellow. We name the Video Swin Transformer trained with risk assessment and PRDeepMellow as PRDeepMellow Swin Transformer.

The Q-target network helps stabilize the training target, improving the overall training process and mitigating the overestimation issue caused by the max operator during greedy policy execution. However, if the Q-target network is removed to improve memory efficiency, it will lead to training instability and the occurrence of overestimation issues. The optimal action-value function is written as:

$$Q^*(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) \max_{a' \in A} Q^*\left(s', a'\right)] \tag{23}$$

where $Q^*(s,a)$ is the optimal action-value function. From Eq. (23), the primary factor contributing to the notable disparity in the Q-network training target is the implementation of a greedy strategy, which causes the target to shift towards the optimal direction with each iteration. Thus, replacing the max operator with one that considers global average information can stabilize training and reduce overestimation.

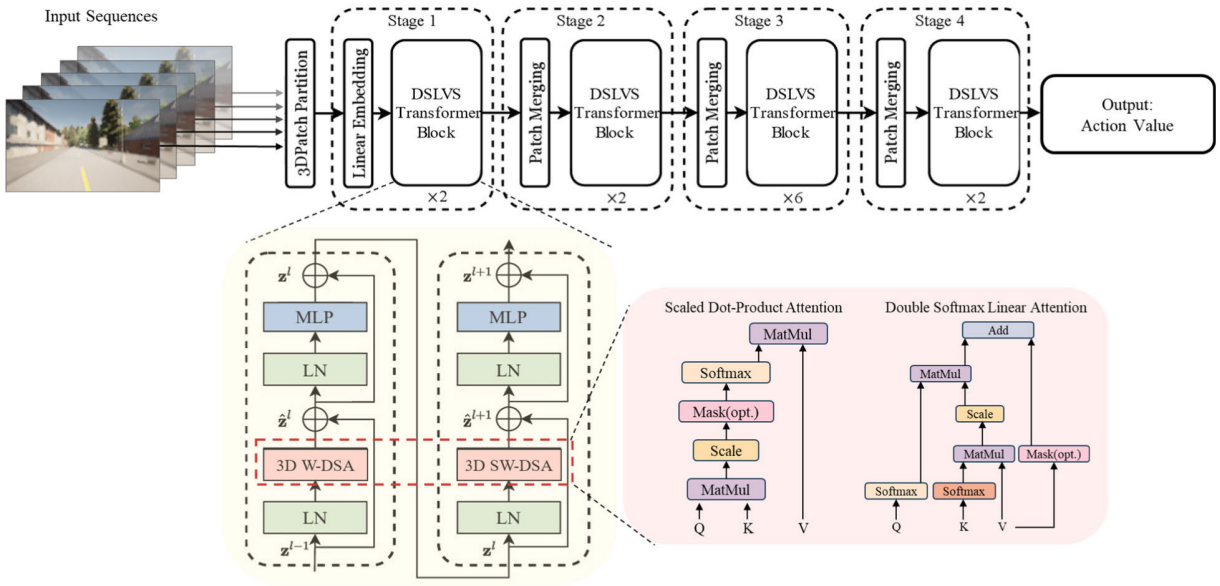


Fig. 6. The overall framework after replacing the vanilla self-attention.

Using the softmax operator combined with global average information can stabilize the training process and help control overestimation. However, it has drawbacks. The softmax operator changes the relative distances between inputs and outputs, smoothing local information. Additionally, its exponential nature can sometimes lead to instability and value explosions.

To tackle the uncertainty and instability caused by the exponential function in the softmax operator, the mellowmax operator has been introduced to improve Eq. (23). The mellowmax operator refines softmax by averaging, applying a logarithm, and adjusting the scale, aligning the scale of the feature space with that of the input. The re-modeled Eq. (23) is shown as:

$$Q^*(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) mm_\omega Q^*\left(s', a', \theta\right)$$
$$mm_\omega = \frac{log\left(\frac{1}{n}\sum_{i=1}^{n} e^{\omega x_i}\right)}{\omega} \tag{24}$$

When the max operator is replaced by the mellowmax operator, the Q-value overestimation is quantified as:

$$\begin{aligned}
&\mathbb{E}\left(mm_\omega\left(\hat{Q}\right) - mm_\omega(Q)\right) = \mathbb{E}\left(mm_\omega\left(\hat{Q}\right)\right) - mm_\omega(Q) \\
&\xrightarrow{\text{Taylor series}} \mathbb{E}\left(\nabla mm_\omega(Q)^T\left(\hat{Q} - Q\right)\right) \\
&+ \frac{1}{2}\mathbb{E}\left(\left(\hat{Q} - Q\right)^T \nabla^2 mm_\omega(Q)\left(\hat{Q} - Q\right)\right) \\
&= \nabla mm_\omega(Q)^T \mathbb{E}\left(\hat{Q} - Q\right) + \frac{1}{2}\nabla^2 mm_\omega \mathbb{E}\left(\left(\hat{Q} - Q\right)^2\right) \\
&= \frac{1}{2}\nabla^2 mm_\omega \cdot \text{var}\left(\hat{Q}\right) \\
&\xrightarrow{\text{Second derivative of mellowmax}} \frac{1}{2}\omega x(1-x) \cdot \text{var}\left(\hat{Q}\right) \\
&where\ x = \frac{e^{\omega Q_i}}{\sum_{i=1}^{n} e^{\omega Q_i}}
\end{aligned} \tag{25}$$

where $\hat{Q}$ is the with error estimates of Q function and Q is the error-free estimation of Q function. It is also assumed that the estimation of the Q function is unbiased, i.e., $\mathbb{E}\left(\hat{Q} - Q\right) = 0$.

Eq. (25) indicates that the extent of overestimation is contingent upon the selection of ω when assuming an accurate Q function estimation. The specific selection of ω is detailed in V-A.

C. Low Latency: Linear Attention for High Inference Speed on Long Sequences

The lightweight decision network structured in a spatial-temporal parallel inference manner serves to reduce the number of parameters while simultaneously increasing the inference speed. However, the low resolution of input video frames limits the extraction of fine details and may cause long-tail effects in real driving scenarios [34]. Increasing video resolution during retraining could mitigate this issue but would slow down inference. This is because the Video Swin Transformer employs vanilla self-attention, which significantly increases computational complexity for long sequences.

To address this issue, the vanilla self-attention is replaced with a linear self-attention, and a decision network that enhances the inference speed for long sequences called Double Softmax Linear Video Swin Transformer (DSLVS Transformer) is proposed in this study. The overarching network architecture is depicted in Fig. 6. The details of this architecture are presented in the following sub-sections.

1) Low-Rank Bottlenecks in Linear Self-Attention: The performance of linear self-attention formed solely by selecting the kernel function tends to degrade significantly [35], [36]. From the perspective of matrix rank, a higher matrix rank is associated with a larger feature space and greater information expression.

Algorithm 3 Computational Complexity of 4 Times Channels Count Linear Self-Attention

Step 1: Project the input X into Q, K, V:
Dimensional Transformation: $n \times hd \bigotimes hd \times 4hd$, Computation: $12n\,(hd)^2$
Step 2: Computation of one Head in Self-Attention:
2-1: Computation of Attention Weight Matrices for Each h Heads $attn = K^T V$:
Dimensional Transformation: $4d \times n \bigotimes n \times 4d$, Computation: $16nd^2$
2-2: Applying Weights to V in Each h Heads:
Dimensional Transformation: $n \times 4d \bigotimes 4d \times 4d$, Computation: $16nd^2$
Total Computation of the Module: $32nhd^2$
Step 3: Fuse Multi-Head Attention through MLP:
Dimensional Transformation: $n \times 4hd \bigotimes 4hd \times hd$, Computation: $4n\,(hd)^2$
Total Computation of the Improved Self-Attention Module: $16n\,(hd)^2 + 32nhd^2$

The attention weight matrix formed by the vanilla self-attention mechanism QK^T has dimensions $n \times n$. Considering that $Q, K \in \mathcal{R}^{n \times d}$ and that the number of channels d is generally much smaller than the sequence length n, the rank of the resulting matrix is limited to d. However, vanilla self-attention includes the softmax operator, which is both computationally expensive and nonlinear, thereby increasing the matrix's rank. This adjustment can move the matrix closer to full rank, thereby improving its effectiveness in processing information.

The linear self-attention operates by first computing $K^T V$, producing a matrix of dimension $d \times d$. However, this process does not involve the application of nonlinear operators like softmax. As a result, the rank of this matrix is generally lower compared to that in vanilla self-attention, leading to a reduced dimensionality of the feature space. Such a reduction may limit the model's effectiveness in processing information. This phenomenon is known as the low-rank bottleneck in linear self-attention. Following the findings of Choromanski et al. [37], the number of channels is increased to four times the original to address the low-rank bottleneck in linear self-attention. The analysis of the corresponding calculated costs is shown in Algorithm 3.

2) Replacing the Vanilla Self-Attention: In the vanilla self-attention, the inputs undergo softmax operations, and the output values sum to 1. Therefore, when selecting a kernel function, it's essential to ensure that the output values are non-negative and to consider the issue of normalization. If Q is normalized in the d dimension and K is normalized in the n dimension, the result of QK^T will also remain normalized. Accordingly, $\phi(\cdot)$ and $\varphi(\cdot)$ are set as two softmax functions, each normalized in certain dimensions:

$$Attention\,(Q, K, V) = Softtmax_2\,(Q)\,Softmax_1\,(K)^T V \tag{26}$$

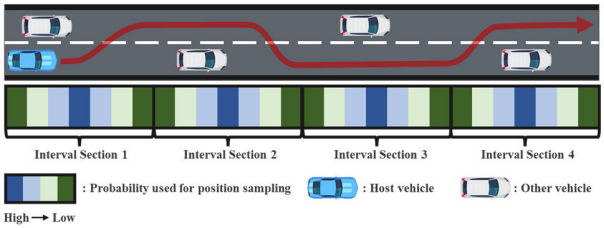


Fig. 7. An illustration for the scenario in our experiments.

where $\phi(\cdot)$ is selected as $Softmax_2$, to indicate normalization along the second dimension and $\varphi(\cdot)$ is selected as $Softmax_1$ to indicate normalization along the first dimension.

In the context of applying a linear self-attention based on double softmax, it is essential to replace the 3D windows attention component while maintaining consistency with the remaining components. The 3D windows attention typically requires the use of relative positional encoding and masking. Both the relative position encoding and the mask are summed at QK^T as follows:

$$Softmax\left(\frac{Q_i K^T}{\sqrt{d_k}} + B\right) V \tag{27}$$

$$Softmax\left(\frac{Q_i K^T}{\sqrt{d_k}} + Mask\right) V \tag{28}$$

Relative position encoding affects only the attention weight matrix, making its separate application to QKV vectors unnecessary in linear self-attention. The aim is to investigate how linear self-attention integrates with the masking mechanism. The function of the mask is to prevent attention from being directed to non-adjacent components. Accordingly, the softmax operator should be applied to the mask without modifying its values. The combination of these two elements is described as follows:

$$\begin{aligned} Softmax\left(\frac{Q_i K^T}{\sqrt{d_k}} + Mask\right) V &= Softmax\left(\frac{Q_i K^T}{\sqrt{d_k}}\right) V \\ &\quad + Softmax\,(mask)\,V \\ &= Softmax\left(\frac{Q_i K^T}{\sqrt{d_k}}\right) V \\ &\quad + mask \times V \\ &\xrightarrow{\text{Linearize}} Softmax_2\,(Q) \\ &\quad \times \frac{Softmax_1\,(K)^T V}{\sqrt{d_k}} \\ &\quad + mask \times V \end{aligned} \tag{29}$$

The Double Softmax Self-Attention (DSA) module is shown as the pink rectangle in Fig. 6. Finally, based on DSA and quadruple channel count (in this study, the raw channel count is 96), the DSLVS Transformer algorithm has been proposed.

IV. Experiment Scenarios

Three lane change scenarios with different traffic environments are produced in CARLA to examine the effectiveness and advances of our proposed method. At the start of each experiment, the ego vehicle begins at rest and accelerates

from the initial section of the road with a throttle input set to 0.5. Given the longest straight road in CARLA is only 420 meters, the experiment is repeated 100 times on this road, with different obstacle scenarios randomly initialized. The model's performance is comprehensively analyzed based on the results of these 100 experiments.

During the experiments, the road is divided into 60-meter segments, with each segment further subdivided into four zones. Vehicles in each zone are positioned based on a Gaussian distribution. Both ends of each segment include lane-changing spaces with a maximum width of 5 meters. To ensure environmental diversity, at least 10 obstacle vehicles are deployed, and the total number of vehicles reaches a maximum of 26 when all zones are filled. A schematic of the scenario is shown below:

The experiment involves three lane-changing scenarios, each with different motion patterns for the other obstacle vehicles. These scenarios are as follows:

(1) Scenario I: Stationary Vehicles - In this scenario, obstacle vehicles are initialized according to the aforementioned motion modes. The autonomous vehicle is required to navigate around all obstacles by changing lanes to complete the driving task in the shortest time possible under safe conditions.

(2) Scenario II: Vehicles Moving at Constant Speed - In this scenario, obstacle vehicles are set to the Autopilot mode, in which the vehicles run with a constant speed of 30m/s without lane change.

(3) Scenario III: Random Lane Change with Speed Variations - This scenario is the most challenging among the three. When the ego vehicle approaches the nearest obstacle vehicle, there is a 50% chance that the obstacle vehicle will change its original motion mode, performing a random lane change, sudden acceleration, or sudden deceleration. The steering value ranges from -1.0 to 1.0, while acceleration varies between 0 and 0.2, and deceleration ranges from -0.1 to 0.

It is important to note that the risk assessment for Scenario III differs from that of Scenarios I and II. In the first two scenarios, obstacle vehicles remain in their original lane without changing lanes, so the ego vehicle only needs to evaluate the risk from vehicles in its current lane. In contrast, Scenario III requires the ego vehicle to assess the risks from vehicles in both lanes simultaneously, making it a more accurate representation of real-world driving conditions.

V. Results and Discussion

A. Determination of Hyperparameter ω

The value of ω is determined during training in Scenario I by conducting four attempts.

Attempt (1): The hyperparameter ω is fixed at a constant value, which is determined through a grid search. For image inputs, some experiments used $\omega = 3000$ [24].

Attempt (2): The hyperparameter ω alternates periodically between 0.01 and 1000, with a higher proportion of instances being assigned the value 1000. This approach aims to alleviate overestimation by using the mean to mitigate it across epochs.

Attempt (3): The hyperparameter ω gradually increases from 0.01 to 1000 as training progresses. Initially, due to poor model stability and a high possibility of overestimation, a smaller ω is used for unbiased estimation. As the model stabilizes, a larger ω is utilized to facilitate effective exploration and reduce overestimation impact.

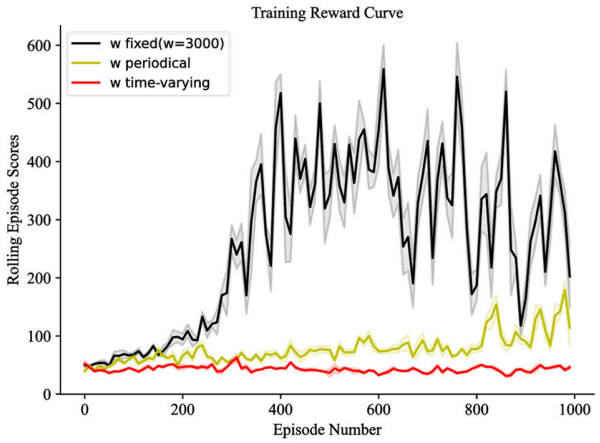


Fig. 8. The reward comparison results obtained during the training process for the examined attempts.

Attempt (4): The hyperparameter ω is piecewise constant, representing a discretized version of Attempt (3). Here, ω is varied uniformly over time, transitioning smoothly to another value after a certain number of training epochs, segmenting the range of ω from 0.01 to 1000 into several levels.

The reward comparison results are shown in Fig. 8. The black, yellow, and red curves represent the fixed value approach (Attempt 1), the periodical changes approach (Attempt 2), and the time-varying changes approach (Attempt 3). However, during the segmented approach (Attempt 4), significant changes in the hyperparameter ω frequently lead to gradient explosions, making this approach unsuitable for training decision networks. Accordingly, the reward curve for Attempt 4 is not illustrated in Fig. 8.

As shown in Fig. 8, the maximal total reward is obtained when the hyperparameter ω is fixed, clearly outperforming the other two training attempts. Additionally, the other two attempts are prone to gradient explosions when the hyperparameter ω changes. Given these observations, fixing the hyperparameter value may be the most effective training approach for PRDeepMellow.

In the above analysis, the hyperparameter ω is treated as a fixed value determined through grid search. Initially, a wide-range search is conducted to establish the approximate range of ω. This range spans from 0.01 to 10,000, with sampling points increasing by a factor of 10 at each step.

As shown in Fig. 9, the total reward associated with the dark green curve is the highest. This indicates that the optimal range for the hyperparameter ω is between 0.01 and 10,000, with 1,000 being the optimal value. Additionally, the Fig. 9 depicts the pattern of total reward in relation to alterations in ω. As the value of ω increases gradually, the total reward initially rises and then declines. This pattern is consistent with the characteristics of the mellowmax operator. When the value of ω is relatively low, the mellowmax operator exhibits a tendency to gravitate towards the mean operator. This prevents the Q-network from effectively exploring optimal decision actions.

TABLE I
COMPARISON OF REAL-TIME EVALUATION METRICS BEFORE AND AFTER LIGHTWEIGHTING

Metrics \ Method	DSCNN Transformer	Video Swin Transformer (RA)	PRDeepMellow Swin Transform	DSLVS Transformer	DSLVS Transformer (scale=4)
Parameter Size (M)	45.0	27.5	27.5	27.5	53.4
Maximum Training Memory Usage (MB)	5607.13	5127.72	5065.60	5129.51	5743.86
FPS (per/s)	34.10	47.11	47.11	47.09	42.38

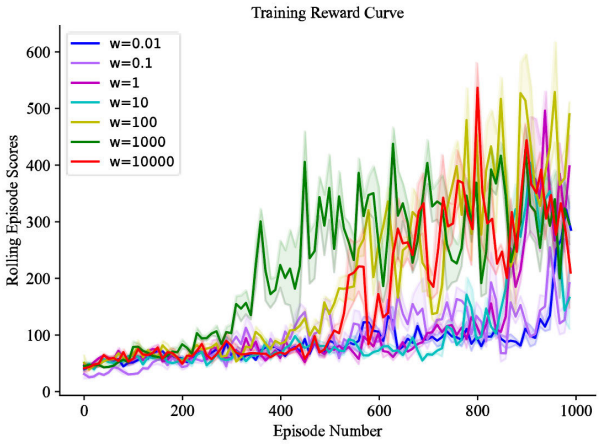


Fig. 9. Grid search for hyperparameter ω with wide sampling range.

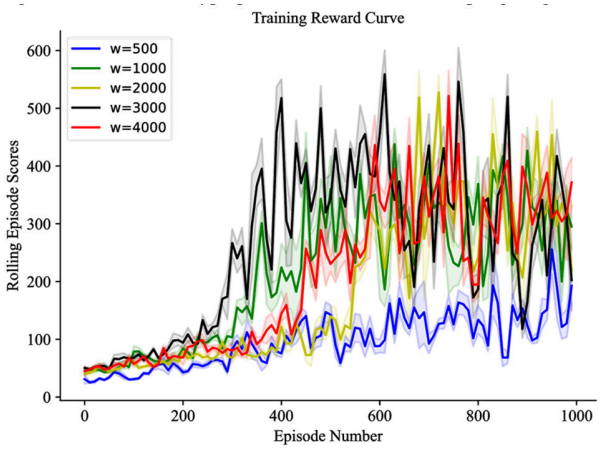


Fig. 10. Grid search for hyperparameter ω with narrow sampling range.

As ω increases beyond a certain range, the tendency towards overestimation becomes more pronounced. This impedes the network's capacity to identify the optimal strategy.

Once the approximate range of the hyperparameter ω has been determined, a more precise sampling interval is employed to ascertain the final value of ω. As illustrated in Fig. 10, the total reward of the black curve is the highest. The optimal value for the hyperparameter ω is identified as 3,000. Accordingly, the PRDeepMellow Swin Transformer is configured with a hyperparameter $\omega = 3,000$.

B. Real-Time Evaluation Results

To evaluate the real-time performance of different algorithms, three metrics are selected for analysis. Parameter size denotes the storage space required for deploying the model, while maximum memory usage indicates the memory space needed during training. Frames per second (FPS) quantifies the number of forward inferences per second, is used to evaluate the inference speed. The results of the real-time evaluation of the different examined algorithms are presented in Table I.

Compared to the DSCNN Transformer, the Video Swin Transformer (RA) achieves a 38.9% reduction in parameter size, an 8.6% decrease in maximum training memory usage, and a 38.2% increase in FPS. The findings illustrate that the utilization of the Video Swin Transformer for the extraction of both temporal and spatial features can effectively reduce the number of model parameters and enhance the inference speed. This indicates that experimentation with the Video Swin Transformer (RA) is a meaningful approach for practical algorithm deployment.

The PRDeepMellow Swin Transformer has improved the training process of the PRDQN algorithm in comparison to the Video Swin Transformer (RA). Consequently, the PRDeepMellow Swin Transformer exhibits the same parameter size and inference speed as the Video Swin Transformer (RA), while the maximum training memory requirement is reduced by 62MB. Furthermore, the algorithm's capacity to perceive subtle environmental changes has been enhanced by improvements in the PRDQN training process. The specific results will be presented in V-C.

To conduct a more comprehensive analysis of the performance of DSLVS Transformer, the real-time evaluation results of DSLVS Transformer and DSLVS Transformer (scale $= 4$) are presented. Outside of this section, DSLVS Transformer refers to DSLVS Transformer (scale $=4$) by default. Additionally, attempts to train the DSLVS Transformer with the improved PRDQN algorithm have not yielded a suitable weight parameter ω that enables optimal performance. As a result, training continues with the original PRDQN algorithm.

In comparison to the Video Swin Transformer (RA), the DSLVS Transformer merely substitutes the native self-attention with linear self-attention. From the results, all three metrics are essentially indistinguishable from those of the pre-improved algorithm. These findings suggest that substituting only the linear self-attention has no effect on the real-time evaluation in this experiment. The DSLVS Transformer (scale $= 4$) improved to mitigate the low-rank

TABLE II
COMPARISON OF PERFORMANCE EVALUATION METRICS BEFORE AND AFTER LIGHTWEIGHTING

Metrics	Method	DSCNN Transformer	Video Swin Transformer (RA)	PRDeepMellow Swin Transformer	DSLVS Transformer
Scenario I	Mean (μ)	360.4	379.2	352.2	203.6
	Variance (σ)	74.9	73.5	130.7	134.6
	nC	18	34	28	15
Scenario II	Mean (μ)	339.8	225.4	349.4	163.2
	Variance (σ)	112.1	146.9	82.9	109.4
	nC	24	66	56	44
Scenario III	Mean (μ)	264.4	337.2	368.5	202.2
	Variance (σ)	141.0	103.7	95.8	99.0
	nC	33	38	22	56

bottle of the linear self-attention has increased the number of parameter size by 25.6M, and the maximum training memory has increased by 12%. This reduction in real-time is still within acceptable limits, given the actual deployment to demand. The experiment conducted on the CARLA simulation platform employs input video frames of low resolution, which impedes the linear self-attention from operating within the optimal range. As a result, compared to the Video Swin Transformer, the FPS of the DSLVS Transformer remains largely unchanged, while the FPS of the DSLVS Transformer (scale = 4) decreases. This is a correct result. The relationship between inference speed and sequence length for the linear attention mechanism will be explored in V -D.

C. Performance Evaluation Results

Lightweight processing often impacts algorithm performance. This section will analyze the performance of various algorithms after lightweight processing to evaluate the resulting changes in their performance.

The maximum distance traveled by the vehicle before a collision or deviation from the lane edge is employed as the fundamental metric for evaluating the efficacy of the algorithm. It is important to note that a longer travel distance indicates the vehicle has successfully completed more lane change. This directly reflects the strength of the algorithm in performing lane change, rather than merely reflecting the length of the traveling distance. Fig. 11, 12, and 13 illustrate the maximum distance traveled by different algorithms in three experimental scenarios during 100 independent experiments. The Score (m) represents the maximum distance traveled. Furthermore, the mean (μ) and variance (σ) of the maximum traveling distance and the number of collisions (nC) are calculated for each of the 100 experiments. These values are then employed as metrics for evaluating the capability, stability, and safety of the algorithm. The specific results are presented in Table II.

To provide a clearer view of the motion trajectory and risk quantification of different algorithms in various scenarios, additional independent tests are performed. Experimental data on the successful operation of each model in different scenarios are obtained. Fig. 14, 15, and 16 illustrate the risk quantification and motion trajectory of different algorithms in three scenarios. As shown in Fig. 14(a), the red rectangular area denotes an elevated risk level for the vehicle in proximity to an obstacle. In the absence of a lane-changing decision,

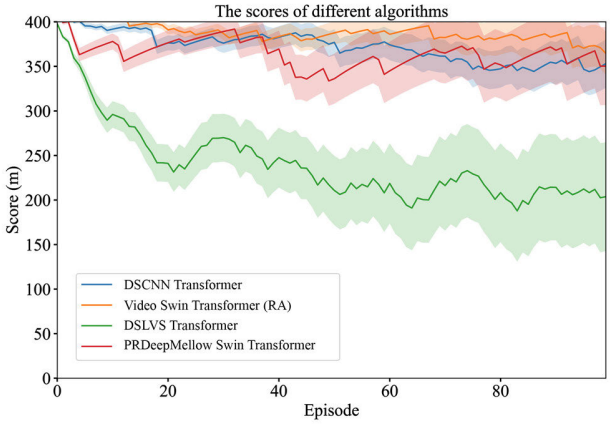


Fig. 11. The scores of algorithms in Scenario I.

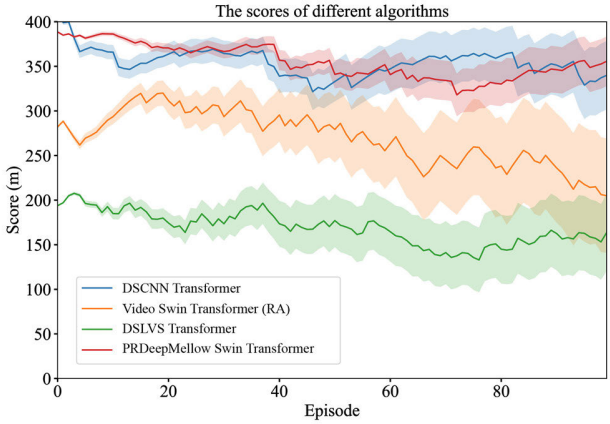


Fig. 12. The scores of algorithms in Scenario II.

a collision is probable. The green rectangular area represents a lower risk level for the vehicle, indicating a state of attentive or safe. The transition from the red to the green area demonstrates that the algorithm is capable of detecting an increase in risk level and making the appropriate decision to reduce it, indicating its ability to recognize and respond to risks.

1) Performance Evaluation Results of Video Swin Transformer (RA): The Video Swin Transformer (RA) demonstrates satisfactory performance across most scenarios. In Scenario I, its performance is comparable to that of the DSCNN Transformer. However, in Scenario II, the model exhibits a noticeable decline in performance. Conversely, in the most

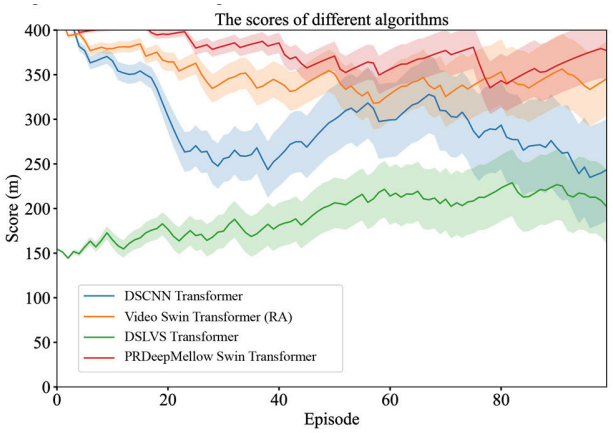


Fig. 13. The scores of algorithms in Scenario III.

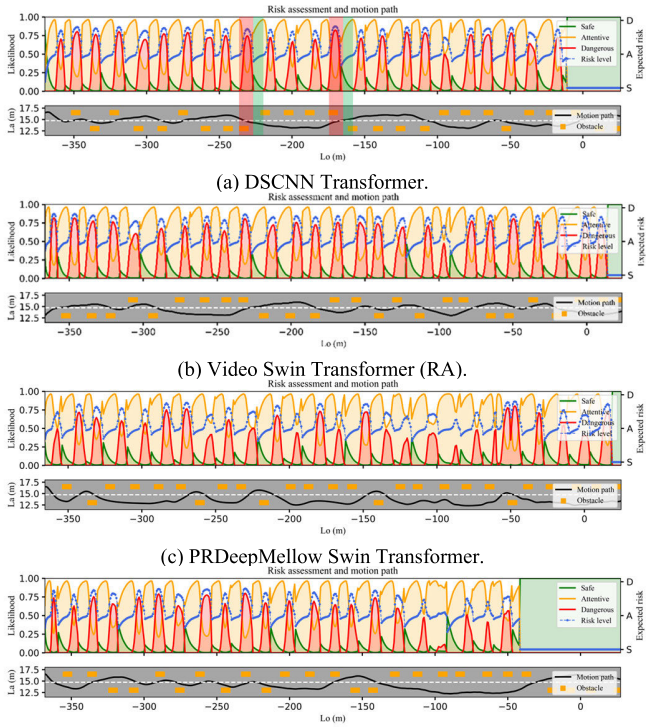


Fig. 14. Risk quantification and motion trajectory of different algorithms in Scenario I.

challenging Scenario III, it demonstrates superior performance. Specifically, in Scenario I, the Video Swin Transformer (RA) scores higher than the DSCNN Transformer in the majority of the 100 independent tests. Corresponding to this, in Table II, the Mean (μ) for the Video Swin Transformer (RA) shows an increase of 5.2% compared to the DSCNN Transformer. In addition, the Variance (σ) of Video Swin Transformer (RA) is also reduced by 1.9% compared to the DSCNN Transformer. These results suggest that both the performance and stability of the model have been enhanced. However, the number of collisions (nC) increases by 16, indicating a decrease in safety. In Scenario II, while the feature extraction of windows attention prioritizes capturing salient global information, it struggles to effectively grasp slowly evolving local details. As a result, while the vehicle navigates

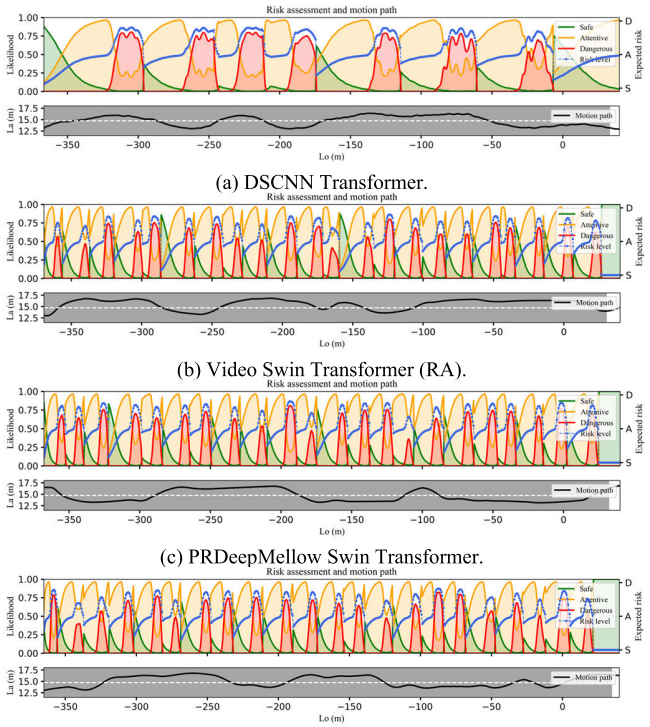


Fig. 15. Risk quantification and motion trajectory of different algorithms in Scenario II.

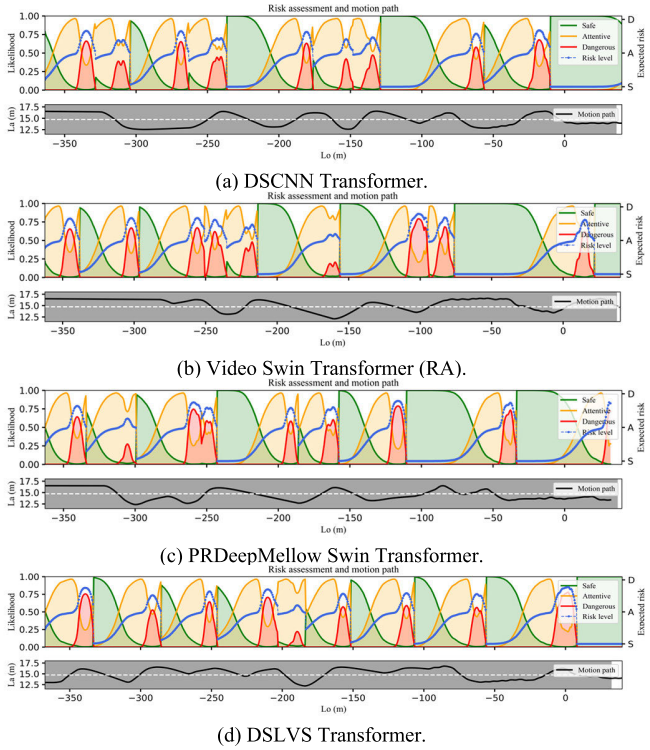


Fig. 16. Risk quantification and motion trajectory of different algorithms in Scenario III.

the 200-300 meters stretch of road, its similar speed to surrounding vehicles often leads to frequent collisions during lane change due to insufficient attention to these subtleties. This directly results in a significant 33.7% reduction in the Mean (μ) and a 31% increase in the Variance (σ). In addition,

the number of collisions (nC) increases by 42. Taken together, these factors significantly

degrade the vehicle's performance in Scenario II. The scenario III is the most challenging among all scenarios. The Video Swin Transformer (RA) exhibits robust performance in this scenario, attributable to the algorithm's capacity to accurately discern and respond to significant alterations in the movement of environmental impediments. As illustrated in Fig. 13, the Video Swin Transformer (RA) is exhibiting superior performance over 100 experiences when compared to the DSCNN Transformer. This is corroborated by the findings in Table II, which indicate a 27.5% increase in the Mean (μ) and a 26.0% reduction in the Variance (σ). Despite an increase of five in the number of collisions (nC), the overall performance remains robust.

As illustrated by the motion trajectory and risk results graphs, the Video Swin Transformer (RA) continues to reduce driving risk through the implementation of rational decision-making processes. In Scenario I, both algorithms demonstrate comparable performance, exhibiting the ability to change lanes correctly, avoid static vehicles, and reach the endpoint. In Scenario II, the Video Swin Transformer is observed to navigate risky driving environments with greater frequency than the DSCNN Transformer. This corresponds to the previous result. In Scenario III, the Video Swin Transformer (RA) exhibits an effective comprehension of the comprehensive risk awareness, an enhanced capacity to discern dynamic content, and a superior performance characterized by reduced driving frequency in high risk level situations and increased frequency in low risk level situations.

In general, the Video Swin Transformer (RA) which has undergone parameter optimization for lightweight performance delivered commendable experimental results.

2) Performance Evaluation Results of PRDeepMellow Swin Transformer: Scenario II is the most commonly encountered situation in real driving, making it crucial for the algorithm to perform well in this context. The PRDeepMellow Swin Transformer employs the non-expansive operator mellowmax, which serves the function of preventing the attention mechanism from becoming sparse. This results in an enhanced algorithmic capacity to perceive local information, markedly improving the algorithm's efficacy in both Scenario II and Scenario III. Specifically, Fig. 12 clearly demonstrates that the PRDeepMellow Swin Transformer exhibits superior performance in terms of the maximum driving distance compared to the Video Swin Transformer (RA). As demonstrated by the data presented in Table II, the PRDeepMellow Swin Transformer has been observed to enhance the Mean (μ) by 55%, diminish the Variance (σ) by 43.6%, and reduce the number of collisions (nC) by 15.2%. In Scenario III, the Mean (μ) is observed to have increased by 9.3%, while the Variance (σ) has decreased by 7.6%.

Additionally, a notable reduction in the number of collisions (nC) is evident, with a 42.1% decline observed. This indicates that the PRDeepMellow Swin Transformer exhibits superior capability, stability, and safety compared to the Video Swin Transformer (RA) in both Scenario II and Scenario III. In Scenario I, the use of a single Q-network for training may result in an overestimation of high-reward actions, such as lane change. Such outcomes may result in unstable driving, lane departure, and experimental failure. This leads to a slight reduction in both capability and stability. In general, the algorithm's performance in Scenario I remains within an acceptable range.

Similar to the capabilities demonstrated by the Video Swin Transformer (RA), the PRDeepMellow Swin Transformer also effectively makes rational decisions that contribute to reduced driving risks. In Scenario I, the partial overestimation problem gives rise to a pronounced jitter phenomenon in the PRDeepMellow Swin Transformer's driving trajectory, with the vehicle traveling in closer proximity to the boundary. This ultimately results in a decline in the model's performance in Scenario I. In both Scenarios II and III, the PRDeepMellow Swin Transformer is observed to maintain smooth trajectories that remain centered on the road.

In general, the PRDeepMellow Swin Transformer reduces maximum training memory usage and addresses the limitations of the Video Swin Transformer (RA) in extracting subtle local variations. The model exhibits robust performance across all three experimental scenarios.

3) Performance Evaluation Results of DSLVS Transformer: The substitution of vanilla self-attention with linear self-attention frequently results in a notable reduction in model performance, predominantly due to the relatively low rank of the linear attention matrix. To circumvent the potential for the model to become constrained by a low-rank bottleneck, the number of feature channels is augmented to a value four times that of the original. However, this modification results in suboptimal performance of the DSLVS Transformer on short-sequence problems within the current experimental setup. The primary challenge is that the input sequence length is insufficient compared to the required number of feature channels necessary for the enhanced algorithm. This leads to an uneven distribution of attention, causing the loss of important features and diminishing the perceptual capability of the DSLVS Transformer. The results of performance evaluation clearly demonstrate this problem.

Firstly, an analysis of the risk quantification and motion trajectory graphs shows that the algorithm can effectively recognize risks, enabling the vehicle to detect elevated threats and take measures to mitigate them. By analyzing the trajectory, it can be observed that the DSLVS Transformer algorithm exhibits a tendency to approach the road edge to a greater extent in Scenario I when compared to the Video Swin Transformer (RA). In Scenarios II and III, the trajectories of the DSLVS Transformer exhibit a notable increase in jitter severity and a greater propensity for larger jitter traces. A significant contributing factor to this phenomenon is the linear attention mechanism's inability to extract essential global features of the vehicle during straight-line travel and lane change when confronted with the context of the inappropriate short sequence problem. The poor performance of the DSLVS Transformer in Scenarios I and II in Table II can be attributed to the increased jerking and the vehicle's tendency to travel in closer proximity to the sidelines.

Table II illustrates that the DSLVS Transformer exhibits a reduced number of collisions in both Scenario I and

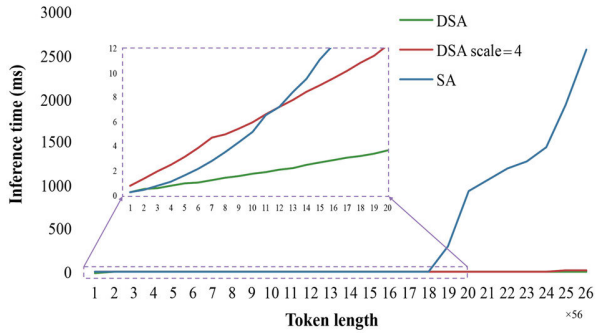


Fig. 17. Variation of inference time with token length.

Scenario II relative to the Video Swin Transformer (RA). However, it achieves a comparatively shorter mean maximum forward distance (μ). This indicates that a greater number of experiments were terminated due to vehicles traversing areas beyond the designated boundaries. The suboptimal performance of the DSLVS Transformer in Scene III can be attributed to the limited focusing capacity of the linear self-attention, which inadequately responds to abrupt changes in the obstacles' movement states. This deficiency is likely to result in an increased number of collisions, thereby degrading overall performance.

Despite the DSLVS Transformer's failure to yield satisfactory outcomes across three experimental conditions, it represents a courageous initiative to tackle lane change decisions with high-resolution visual modal inputs. Future efforts will concentrate on enhancing the feature extraction capabilities and the focus accuracy of the linear self-attention across diverse sequences. Additionally, these improvements will be evaluated using high-resolution image inputs.

D. Analysis of Mechanism Behind Linear Self-Attention

The inferential time trends for varying sequence lengths of the three attention mechanisms, Self-Attention (SA), Double Softmax Linear-Attention (DSA), and Double Softmax Linear-Attention (scale =4) (DSA scale =4), are shown in Fig. 17. The base sequence length is 56, and the base number of channels is 96, which aligns with the experimental component of the setup presented in this study. It illustrates that as the number of tokens increases, the inference time of the SA exhibits a markedly accelerated growth rate in comparison to that of linear self-attention. However, the computational complexity of SA is not initially $O\left(n^2\right)$. The total computational cost of SA is the sum of linear and quadratic terms, specifically $4n\,(hd)^2+2n^2hd$. When $4n\,(hd)^2 < 2n^2hd$, i.e., when $n > 2hd$ (where $n > 192$ in the experiment), the quadratic term becomes dominant.

Examining the detail enlargements reveals that DSA (scale =4) has a longer initial inference time than SA. As previously demonstrated in Algorithm 3, the computational cost is $16n\,(hd)^2+32nhd^2$. When $16n\,(hd)^2+32nhd^2 < 4n\,(hd)^2+2n^2hd$, i.e., when $n > 16d+6hd$ (where $n > 1088$ in the experiment), the computational complexity of DSA (scale = 4) becomes lower than that of SA. Consequently, when the growth factor reaches 12 (theoretically 19.43), the curve for SA begins to exceed that of DSA (scale = 4). At this point, the benefit of linear self-attention for handling long sequences becomes apparent.

TABLE III
COMPARISON OF INFERENCE TIME FOR THREE METHODS

Method	Inference Time (ms)	Rate of Relative Change $\nabla_s(\%)$
SA	0.251	-
DSA	0.276	9.84↑
DSA scale=4	0.796	217.09↑

The data in Algorithm 2 shows that the computational complexity of DSA is $4n\,(hd)^2+2nhd^2$. Under the experimental conditions set in this study, its computational complexity is consistently lower than $n\,(hd)^2+2n^2hd$. Therefore, the inference time of DSA should theoretically always be less than that of SA. However, in practice, this is not the case. Table III presents a comparison of the inference times for the three self-attention mechanisms at $n = 56$.

A comparison of the data in Table III shows that the reasoning time for DSA is longer than that for SA. This discrepancy can be attributed to the additional computation required for the $mask\times V$ operation in the DSA process. In particular, when the sequence length is relatively short, the additional computation can cause the inference time of DSA to exceed that of SA. Nevertheless, as the sequence length increases and the mask matrix becomes increasingly sparse, the computational burden of this component is reduced to a level that can be considered negligible.

In addition to the computational complexity of DSA impacting the Transformer, the feed-forward network (FFN) also plays a significant role in influencing the Transformer's inference speed. The FFN consists of two layers: an initial expansion in dimensionality followed by a reduction. The dimensionality transitions from $n\times hd \bigotimes hd\times 4hd$ to $n\times 4hd \bigotimes 4\ hd\times hd$, leading to a total computational cost of $8n\,(hd)^2$. In the Transformer, the DSA dominates for SA only when $8n\,(hd)^2 < 4n\,(hd)^2+2n^2hd$, i.e., when $n > 2hd$ (in experiments, $n > 192$).

Removing the FFN module would exacerbate the low-rank phenomenon caused by the stacking of multiple self-attention layers. In Transformers, the FFN module works in conjunction with the residual structure to play a crucial role in mitigating induction bias. This combination prevents the feature space from collapsing into a rank-1 space at an exponential rate [38]. While enhancing self-attention for short sequences does not substantially improve the overall inference speed of the decision model, evaluating its performance in such sequences is a relatively low-cost approach. This opens the door to further exploration of more effective linear self-attention mechanisms.

E. Limitations and Future Work

While the proposed lightweight method offers significant advancements, it also has limitations. First, although the PRDeepMellow Swin Transformer reduces memory usage during operation, the experience replay buffer remains the primary source of memory consumption during training, with

no efficient solutions currently available to minimize its usage without affecting performance. Second, the hyperparameter ω in the mellowmax operator is critical for optimization, as its optimal value depends on the perception-layer algorithm and may need adjustments during training. Developing efficient methods for tuning ω is a promising direction for future research. Lastly, while the DSLVS Transformer enhances inference speed for long-sequence tasks, its performance on short-sequence tasks declines. In addition to designing an effective kernel function for short-sequence linear attention, incorporating convolutional operators could enhance the model's ability to capture local patterns and fine-grained information. This hybrid approach could address short-sequence limitations while preserving the model's lightweight design. Future work will focus on optimizing the replay buffer to reduce memory usage and enhancing linear attention performance.

VI. Conclusion

In this study, innovative solutions are provided for the practical deployment of decision-making algorithms in autonomous vehicles through lightweight optimizations from three perspectives: parameter size, training memory usage, and inference speed. A lane change decision-making model is proposed that combines the lightweight network Video Swin Transformer with a risk assessment-based PRDQN algorithm. This process yields a neural network with risk-aware capabilities, designated as the Video Swin Transformer (RA). Based on this model, improvements in training memory usage and inference speed are achieved, leading to the development of PRDeepMellow Swin Transformer and DSLVS Transformer algorithms. The experiments compared DSCNN Transformer with our proposed methods in three lane change scenarios with varying difficulties. The real-time evaluation results and performance evaluation results demonstrate that the Video Swin Transformer (RA) and PRDeepMellow Swin Transformer exhibit commendable driving performance while maintaining a lightweight structure. Additionally, the DSLVS Transformer demonstrates the capability to reduce the complexity of the attentional mechanism from $O\left(n^2\right)$ to $O\left(n\right)$ in long sequence tasks, which laying the groundwork for decision-making algorithms to maintain fast inference speed even with high-resolution images. The contributions in this study are expected to provide feasible solutions to the practical applications of reinforcement learning methods in autonomous vehicles and intelligent transportation systems.

References

[1] P. Hang, Y. Zhang, N. de Boer, and C. Lv, "Conflict resolution for connected automated vehicles at unsignalized roundabouts considering personalized driving behaviours," *Green Energy Intell. Transp.*, vol. 1, no. 1, Jun. 2022, Art. no. 100003.

[2] D. Li, A. Liu, H. Pan, and W. Chen, "Safe, efficient and socially-compatible decision of automated vehicles: A case study of unsignalized intersection driving," *Automot. Innov.*, vol. 6, no. 2, pp. 281–296, May 2023.

[3] M. Fu et al., "Cooperative decision-making of multiple autonomous vehicles in a connected mixed traffic environment: A coalition game-based model," *Transp. Res. C, Emerg. Technol.*, vol. 157, Dec. 2023, Art. no. 104415.

[4] S. Heshami and L. Kattan, "Towards self-organizing connected and autonomous vehicles: A coalitional game theory approach for cooperative lane-changing decisions," *Transp. Res. C, Emerg. Technol.*, vol. 166, Sep. 2024, Art. no. 104789.

[5] G. Yu, H. Li, Y. Wang, P. Chen, and B. Zhou, "A review on cooperative perception and control supported infrastructure-vehicle system," *Green Energy Intell. Transp.*, vol. 1, no. 3, Dec. 2022, Art. no. 100023.

[6] J. Liu, H. Wang, L. Peng, Z. Cao, D. Yang, and J. Li, "PNNUAD: Perception neural networks uncertainty aware decision-making for autonomous vehicle," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 24355–24368, Dec. 2022.

[7] G. Li et al., "Risk assessment based collision avoidance decision-making for autonomous vehicles in multi-scenarios," *Transp. Res. C, Emerg. Technol.*, vol. 122, Jan. 2021, Art. no. 102820.

[8] G. Li, Y. Yang, S. Li, X. Qu, N. Lyu, and S. E. Li, "Decision making of autonomous vehicles in lane change scenarios: Deep reinforcement learning approaches with risk awareness," *Transp. Res. C, Emerg. Technol.*, vol. 134, Jan. 2022, Art. no. 103452.

[9] K. Ji, N. Li, M. Orsag, and K. Han, "Hierarchical and game-theoretic decision-making for connected and automated vehicles in overtaking scenarios," *Transp. Res. C, Emerg. Technol.*, vol. 150, May 2023, Art. no. 104109.

[10] X. He and C. Lv, "Towards safe autonomous driving: Decision making with observation-robust reinforcement learning," *Automot. Innov.*, vol. 6, no. 4, pp. 509–520, Nov. 2023.

[11] S. Xu, Q. Liu, Y. Hu, M. Xu, and J. Hao, "Decision-making models on perceptual uncertainty with distributional reinforcement learning," *Green Energy Intell. Transp.*, vol. 2, no. 2, Apr. 2023, Art. no. 100062.

[12] J. Duan et al., "Encoding distributional soft actor-critic for autonomous driving in multi-lane scenarios [research frontier]," *IEEE Comput. Intell. Mag.*, vol. 19, no. 2, pp. 96–112, May 2024.

[13] G. Ma et al., "Joint partial offloading and resource allocation for vehicular federated learning tasks," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 8, pp. 8444–8459, Aug. 2024.

[14] M. Henning, J. Müller, F. Gies, M. Buchholz, and K. Dietmayer, "Situation-aware environment perception using a multi-layer attention map," *IEEE Trans. Intell. Vehicles*, vol. 8, no. 1, pp. 481–491, Jan. 2023.

[15] W. Hua, Y. Zhou, C. M. D. Sa, Z. Zhang, and G. E. Suh, "Channel gating neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 1886–1896.

[16] Y. Rao, W. Zhao, B. Liu, J. Lu, J. Zhou, and C. J. Hsieh, "DynamicViT: Effcient vision transformers with dynamic token sparsification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 13937–13949.

[17] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," 2015, *arXiv:1510.00149*.

[18] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[19] Z. Hechen, W. Huang, and Y. Zhao, "ViT-LSLA: Vision transformer with light self-limited-attention," 2022, *arXiv:2210.17115*.

[20] S. Venkataramanan, A. Ghodrati, Y. M. Asano, F. Porikli, and A. Habibian, "Skip-attention: Improving vision transformers by paying less attention," 2023, *arXiv:2301.02240*.

[21] G. Fang, X. Ma, M. Song, M. Bi Mi, and X. Wang, "DepGraph: Towards any structural pruning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Vancouver, BC, Canada, Jun. 2023, pp. 16091–16101.

[22] C. Li, G. Wang, B. Wang, X. Liang, Z. Li, and X. Chang, "Dynamic slimmable network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Nashville, TN, USA, Jun. 2021, pp. 8603–7613.

[23] K. Xu, X. Xiao, J. Miao, and Q. Luo, "Data driven prediction architecture for autonomous driving and its application on Apollo platform," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Las Vegas, NV, USA, Oct. 2020, pp. 175–181.

[24] S. Kim, K. Asadi, M. Littman, and G. Konidaris, "DeepMellow: Removing the need for a target network in deep Q-learning," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Macao, China, Aug. 2019, pp. 2733–2739.

[25] T. Qin, Y. Zheng, T. Chen, Y. Chen, and Q. Su, "A light-weight semantic map for visual localization towards autonomous driving," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Xi'an, China, May 2021, pp. 11248–11254.

[26] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[27] M. Hessel et al., "Rainbow: Combining improvements in deep reinforcement learning," in *Proc. AAAI*, Apr. 2018, vol. 32, no. 1, pp. 3215–3222.

[28] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 7794–7803.
[29] Y.-H. H. Tsai, S. Bai, M. Yamada, L.-P. Morency, and R. Salakhutdinov, "Transformer dissection: A unified understanding of transformer's attention via the lens of kernel," 2019, *arXiv:1908.11775*.
[30] S. Zhuoran, Z. Mingyuan, Z. Haiyu, Y. Shuai, and L. Hongsheng, "Efficient attention: Attention with linear complexities," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Waikoloa, HI, USA, Jan. 2021, pp. 3530–3538.
[31] G. Li et al., "Lane change strategies for autonomous vehicles: A deep reinforcement learning approach based on transformer," *IEEE Trans. Intell. Vehicles*, vol. 8, no. 3, pp. 2197–2211, Mar. 2023.
[32] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Montreal, QC, Canada, Oct. 2021, pp. 9992–10002.
[33] Z. Liu et al., "Video Swin transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, New Orleans, LA, USA, Jun. 2022, pp. 3192–3201.
[34] K. Yang, B. Li, W. Shao, X. Tang, X. Liu, and H. Wang, "Prediction failure risk-aware decision-making for autonomous vehicles on signalized intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 11, pp. 12806–12820, Nov. 2023.
[35] S. Bhojanapalli, C. Yun, A. S. Rawat, S. J. Reddi, and S. Kumar, "Low-rank bottleneck in multi-head attention models," in *Proc. Int. Conf. Mach. Learn.*, Jan. 2020, pp. 864–873.
[36] D. Han, X. Pan, Y. Han, S. Song, and G. Huang, "FLatten transformer: Vision transformer using focused linear attention," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Paris, France, Oct. 2023, pp. 5938–5948.
[37] K. Choromanski et al., "Rethinking attention with performers," 2020, *arXiv:2009.14794*.
[38] Y. Dong, J.-B. Cordonnier, and A. Loukas, "Attention is not all you need: Pure attention loses rank doubly exponentially with depth," in *Proc. Int. Conf. Mach. Learn.*, Jan. 2021, pp. 2793–2803.

Guofa Li (Senior Member, IEEE) received the Ph.D. degree in mechanical engineering from Tsinghua University, China, in 2016. He is currently a Professor with Chongqing University, China. His research interests include environment perception, driver behavior analysis, and smart decision-making based on artificial intelligence technologies in autonomous vehicles and intelligent transportation systems. He serves as an Associate Editor for IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON AFFECTIVE COMPUTING, and IEEE SENSORS JOURNAL.

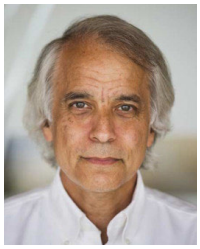
Jun Yan received the B.E. degree from Jiangsu University, Zhenjiang, China, in 2024. He is currently pursuing the M.S. degree with the College of Mechanical and Vehicle Engineering, Chongqing University, Chongqing, China. His research interests include deep reinforcement learning, decision making, and control of intelligent vehicles.

Yifan Qiu received the B.E. degree from Shenzhen University, China, in 2021, where he is currently pursuing the master's degree with the College of Mechatronics and Control Engineering. His research interests include deep reinforcement learning technologies for the development of autonomous vehicles.

Qingkun Li (Member, IEEE) received the Ph.D. degree in mechanical engineering from Tsinghua University, Beijing, China, in 2023. He is currently an Assistant Research Fellow with the Institute of Software, Chinese Academy of Sciences, Beijing. His research interests include driving behavior analysis and human–machine interaction.

Jie Li received the Ph.D. degree in mechanical engineering from Tsinghua University, Beijing, China, 2024. He is currently an Associate Professor with the College of Mechanical and Vehicle Engineering, Chongqing University, Chongqing, China. His research interests include model predictive control, adaptive dynamic programming, and reinforcement learning.

Shengbo Eben Li (Senior Member, IEEE) received the M.S. and Ph.D. degrees from Tsinghua University in 2006 and 2009, respectively. Before joining Tsinghua University, he was with Stanford University, the University of Michigan, and UC Berkeley. He is currently a Professor leading the Intelligent Driving Laboratory (iDLab), Tsinghua University. His active research interests include intelligent vehicles and driver assistance systems, reinforcement learning and optimal control, and distributed control and estimation. He serves as the Board of Governor for IEEE ITS Society and an Associate Editor for IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE TRANSACTIONS ON INTELLIGENT VEHICLES, and *IEEE Intelligent Transportation Systems Magazine*.



Paul Green received the joint Ph.D. degree in industrial and operations engineering and psychology from the University of Michigan in 1979. He is currently a Research Professor with the University of Michigan Transportation Research Institute (UMTRI), Ann Arbor, MI, USA, where he is an Adjunct Professor with the Department of Industrial and Operations Engineering. His research interests include driving safety, driver interfaces, driver behavior, driver workload, and the development of standards to get research into practice. He is a Past President of the Human Factors and Ergonomics Society.