



# ST-MAE: robust lane detection in continuous multi-frame driving scenes based on a deep hybrid network

Rongyun Zhang<sup>1</sup> · Yufeng Du<sup>1</sup> · Peicheng Shi<sup>2</sup> · Lifeng Zhao<sup>3</sup> · Yaming Liu<sup>1</sup> · Haoran Li<sup>1</sup>

Received: 27 April 2022 / Accepted: 25 October 2022 / Published online: 24 November 2022  
© The Author(s) 2022

## Abstract

Lane detection is one of the key techniques to realize advanced driving assistance and automatic driving. However, lane detection networks based on deep learning have significant shortcomings. The detection results are often unsatisfactory when there are shadows, degraded lane markings, and vehicle occlusion lanes. Therefore, a continuous multi-frame image sequence lane detection network is proposed. Specifically, the continuous six-frame image sequence is input into the network, in which the scene information of each frame image is extracted by an encoder composed of Swin Transformer blocks and input into the PredRNN. Continuous multi-frame of the driving scene is modeled as time-series by ST-LSTM blocks, and then, the shape changes and motion trajectory in the spatiotemporal sequence are effectively modeled. Finally, through the decoder composed of Swin Transformer blocks, the features are obtained and reconstructed to complete the detection task. Extensive experiments on two large-scale datasets demonstrate that the proposed method outperforms the competing methods in lane detection, especially in handling difficult situations. Experiments are carried out based on the TuSimple dataset. The results show: for easy scenes, the validation accuracy is 97.46%, the test accuracy is 97.37%, and the precision is 0.865. For complex scenes, the validation accuracy is 97.38%, the test accuracy is 97.29%, and the precision is 0.859. The running time is 4.4 ms. Experiments are carried out based on the CULane dataset. The results show that, for easy scenes, the validation accuracy is 97.03%, the test accuracy is 96.84%, and the precision is 0.837. For complex scenes, the validation accuracy is 96.18%, the test accuracy is 95.92%, and the precision is 0.829. The running time is 6.5 ms.

**Keywords** Lane detection · Deep learning · Swin transformer · Continuous multi-frame

✉ Rongyun Zhang  
hanfengzhiwei@163.com

Yufeng Du  
478859711@qq.com

Peicheng Shi  
shipeicheng@126.com

Lifeng Zhao  
zhao.lin.feng@163.com

Yaming Liu  
1873104922@qq.com

Haoran Li  
314267199@qq.com

<sup>1</sup> The School of Mechanical and Automotive Engineering, Anhui Polytechnic University, Beijing Middle Road, Wuhu 241000, China

<sup>2</sup> Automotive New Technology Anhui Engineering and Technology Research Center, Anhui Polytechnic University, Beijing Middle Road, Wuhu 241000, China

## Introduction

In recent years, with the rapid development of deep learning and computer vision techniques, the autonomous vehicle technique has become possible. In this setting, lane detection is the core of the Advanced Driver Assistance System (ADAS) to realize the active safety control of vehicle's lateral movement. It is thus one of the core techniques of autonomous vehicles. There are many types of lane detection networks, which are mainly divided into two categories, namely, traditional networks and networks based on deep learning. Traditional networks use geometric models to represent lanes and energy minimization to formulate lanes. On the other hand, networks based on deep learning mostly adopt supervised learning strategies to segment lanes. In general,

<sup>3</sup> The School of Automotive and Traffic Engineering, Hefei University of Technology, No.193 Tunxi Road, Hefei 230009, China

networks based on deep learning have shown excellent performance in solving many computer vision tasks (such as object detection [1–3], classification [4–6], and segmentation [7–9]). Deep neural networks are mainly divided into two categories. The first one is the deep convolution neural network (DCNN), which processes scene information through convolution modules of each layer and has an excellent effect in feature extraction of image sequences and video sequences. The second is the deep recurrent neural network (DRNN), which divides the scene information in a multi-frame image sequence into continuous blocks to complete the state propagation to recurrently process the input signal and complete the spatiotemporal sequence task. Most of these networks train the network and complete the lane detection task through a frame of the image sequence. The limited scene information provided by is not enough for accurate lane detection or prediction. Generally, the lanes are continuous structures, and the adjacent frame image sequence can be fused with the features extracted from the current frame image sequence. Hence, we propose a network that uses multi-frame image sequence feature fusion for lane detection. Due to the use of more scene information, the robustness of the network is high, and it can complete the lane detection task in highly challenging scenes. Therefore, we begin to investigate a network utilizing continuous multi-frame image sequences to complete the lane detection task. In this task, the continuous driving scene images have obvious time-series features, but the original image sequence of each frame is used as input, and the network needs to hold many calculation sequences. Considering the vigorous development of transformer [10–14] and computer vision technology in recent years, to reduce the computational cost and to ensure that the lane detection task can be efficiently completed, we propose a hybrid depth network composed of Swin Transformer and Predictive Recurrent Neural Network (PredRNN) [15] based on the MAE [16] network architecture for lane detection of continuous multi-frame image sequences. The network takes a continuous multi-frame image sequence as input and completes the lane detection task by semantic segmentation. It contains an encoder network and a decoder network to ensure that the final output mapping is of the same size as the input. The PredRNN network is used to fuse the scene information in the continuous multi-frame image sequence, which is then used to process the time-series of encoding features, and then input into the decoder network to finally complete the lane detection task.

The main contributions of this paper are as follows:

When there are serious shadow sections, sections with lane marking degradation, and vehicles blocking lanes, the lane detection task cannot be completed using only the current frame image sequence. To get around this, we propose a new network for completing lane detection using continuous multi-frame image sequences. Since multi-frame continuous

images can extract more scene information, the proposed network can predict lanes more accurately and has high robustness.

Second, the hybrid deep network proposed in this paper is different from other deep neural networks. The convolution module is replaced by the pure self-attention mechanism, and the effect of feature extraction by the convolution module is achieved in a way that even better than what is by the convolution module. The network adopts the encoder–decoder architecture where the encoder and decoder are composed of Swin Transformer blocks, and PredRNN is integrated between the encoder and the decoder. Through experiments, it is found that Spatiotemporal LSTM (ST-LSTM) in PredRNN is very effective for the prediction of time-series information, so that the lane detection task can be better completed.

Finally, we refer to the natural language processing (NLP) method and design the encoder and decoder in the MAE asymmetrically. This is conducive for reducing the computational cost. At the same time, our network reduces the size of the multi-frame continuous image sequence through mask technology and still can complete the detection task well.

The structure of this paper is as follows: The section “[Related work](#)” reviews the related work. The section “[Method](#)” introduces the proposed hybrid deep neural network, including the MAE network structure, PredRNN network, masking technique, and training strategy. The section “[Experiment and results](#)” presents the experiments and analyzes the relevant experimental results. Finally, the section “[Discussion](#)” summarizes our work and briefly discusses future research directions.

## Related work

In recent years, a good deal of research has been carried out on lane detection [17–21]. The resulting networks can be divided into two categories: traditional networks and deep learning networks. In this section, we will briefly review these two types of networks.

### Traditional networks

In the lane detection task, the vehicle sensor is used to obtain the geometric feature information of the object. Lane geometric modeling is carried out by curve detection or fitting, and the lane detection task is completed with the help of gradient, color, and texture information. Borkar et al. [22] first detected the lane edge and then realized lane detection by curve fitting. Mohamed Aly [23] used a Gaussian filter to extract the edge features of lanes and completed lane edge detection by calculating the edge features in aerial views.

McCall and Trivedi [24] designed a steerable filter for robust and accurate detection of lane markings, such as circular reflectors, solid lines, and segmented lines under different lighting and road conditions. This helped to improve the robustness of lane detection in the presence of complex shadows, overpasses, tunnel lighting changes, and road changes. Zhou et al. [25] proposed a robust lane detection network based on a geometric model and Gabor filter. After the camera is installed and fixed on the vehicle, it only runs once, and the camera parameters are accurately estimated by 2D calibration. The starting position, original lane direction, and lane width are estimated by dominant direction estimation and local Hough transform. Then, the candidate lane model is constructed to match the final lane model. Selver et al. [26] present a robust approach that partitions a video frame into four regions, each of which is filtered by two-dimensional Gabor wavelets at different scales to complete the lane detection task. Zheng et al. [27] proposed a method to select points in Hough space that are in line with lane parallel features, length and angle features, and intercept features, and directly then converted into a lane equation. Hur et al. [28] used a conditional random field (CRF) to find the best association of multilane markers in complex and challenging urban roads and completed the lane detection task. Although the above-mentioned traditional networks are relatively simple and have good detection effect in the case of good vehicle conditions, they usually have poor detection effect and low robustness in complex scenes (such as shadow occlusion area, lane marking missing, etc.). Therefore, various lane detection networks based on deep learning have been proposed.

### Networks based on deep learning

In recent years, with the rise of deep learning and its successful application in the field of computer vision, many lane detection networks based on deep learning have been proposed. Such networks can be divided into four categories.

#### CNN based on encoder–decoder architecture

This network is often used in semantic segmentation tasks [29, 30]. Wang et al. [31] proposed a network named LaneNet. First, the lane edge suggestion network was used to classify the pixel-level lane edge, and then, the lane edge suggestion was used to detect the lane through the lane localization network.

#### Networks based on FCN architecture

The optimized network framework based on FCN is also widely used in lane detection tasks. Liu et al. [32] proposed an end-to-end network, which directly outputs the parameters of the lane network and uses the network with

Transformer to learn more abundant structures and contexts to complete lane detection. Tabelini et al. [33] proposed a deep lane detection network called LaneATT, which uses anchor points based on an attention mechanism that aggregates global information to complete lane detection tasks. Hou et al. [34] proposed a lane detection network called self-attention distillation (SAD), which allows the network to make substantial improvements using its own learning ability without any additional supervision or labels. Qin et al. [35] proposed a simple but effective network by regarding lane detection as a row-based selection problem with global characteristics. Through row-based selection, the computational cost can be significantly reduced. Moreover, they also proposed a structural loss to explicitly model the structure of lanes. Using a large receptive field on global features. Challenging scenarios could also be handled. Lo et al. [36] proposed two techniques, namely, Feature Size Selection (FSS) and Degressive Dilation Block (DD Block). The FSS allows a network to extract thin lane features using appropriate feature sizes. To acquire fine-grained spatial information, the DD block is made of a series of dilated convolutions with degressive dilation rates.

#### Networks based on CNN+RNN architecture

Li et al. [37] proposed a network combining Convolution Neural Network (CNN) and Recurrent Neural Network (RNN). This network can simultaneously detect the existence of the target and the geometric properties (position and direction) of the target relative to the region of interest, and the recursive neuron layer can be used for structured visual detection. A multitask convolutional neural network provides auxiliary geometric information to facilitate the subsequent modeling of a given lane structure. The recursive neural network automatically detects the lane boundary, including the area without any marks, and can complete the lane detection task without any clear prior knowledge or secondary modeling.

#### Networks based on GAN architecture

A generative adversarial network (GAN) [38] is composed of a generator and an identifier, which can be used for lane detection. Ghafoorian et al. [39] proposed EL-GAN, a GAN framework, to mitigate the discussed problem using an embedding loss. With EL-GAN, one can discriminate based on learned embeddings of both the labels and the prediction at the same time. This results in much more stable training due to having better discriminative information, benefiting from seeing both ‘fake’ and ‘real’ predictions at the same time. This substantially stabilizes the adversarial training process.

Although the performance of the lane detection networks based on deep learning proposed above is significantly

improved compared with the traditional lane detection network, the computational cost is greatly increased. Moreover, in the process of feature extraction, the scene information will be missing due to the feature extraction of a single-frame image. This greatly hinders the improvement of network performance through pretraining. To solve the above shortcomings, this paper proposes robust lane detection method for continuous multi-frame driving scenes based on a deep hybrid network.

## Method

### System overview

A lane is composed of solid or dotted lines on the road. The lane detection task can be completed by geometric modeling or semantic segmentation based on the scene information in an image. However, due to the poor robustness of these networks under challenging scenarios, such as severe shadow, mark degradation, and vehicle occlusion, the networks reviewed above cannot be applied to the actual ADAS system. Furthermore, the information present in a single image is not sufficient to support robust lane detection. In fact, in the actual driving scene, the images captured by the vehicle camera are continuous. Hence, the lanes in the current frame image sequence usually overlap with those in the previous frame. This makes it possible to detect the lanes via the spatiotemporal sequence prediction. To solve the problem of lane detection in challenging scenarios, PredRNN is very suitable for multi-frame image lane detection tasks due to its ability to perform continuous signal processing, sequence feature extraction, and integration. The network architecture based on the MAE encoder–decoder has great advantages in dealing with large-scale image tasks. Therefore, the integration of the two greatly enhances the ability of the network in semantic segmentation. We propose a new deep hybrid network that takes the masked autoencoders (MAE) as the main framework and integrates the predictive recurrent neural network (PredRNN). The detection task of a continuous multi-frame image sequence of a driving scene can then be realized.

The improved MAE encoder–decoder network and PredRNN network are integrated into an end-to-end trainable network. The network architecture is shown in Fig. 1. First, each input image is patched out. Then, we mask a large quantity of random image block subsets out in each image. A small portion of the visible patch is utilized in the encoder. The mask token is introduced after the encoder. The full set of encoded patches and mask tokens are processed by a small decoder, which reconstructs the original image in pixels. Finally, the decoder is discarded, and the encoder utilizes the uncorrupted images to produce a representation for the recognition task.

## Network design

### Optimized MAE network

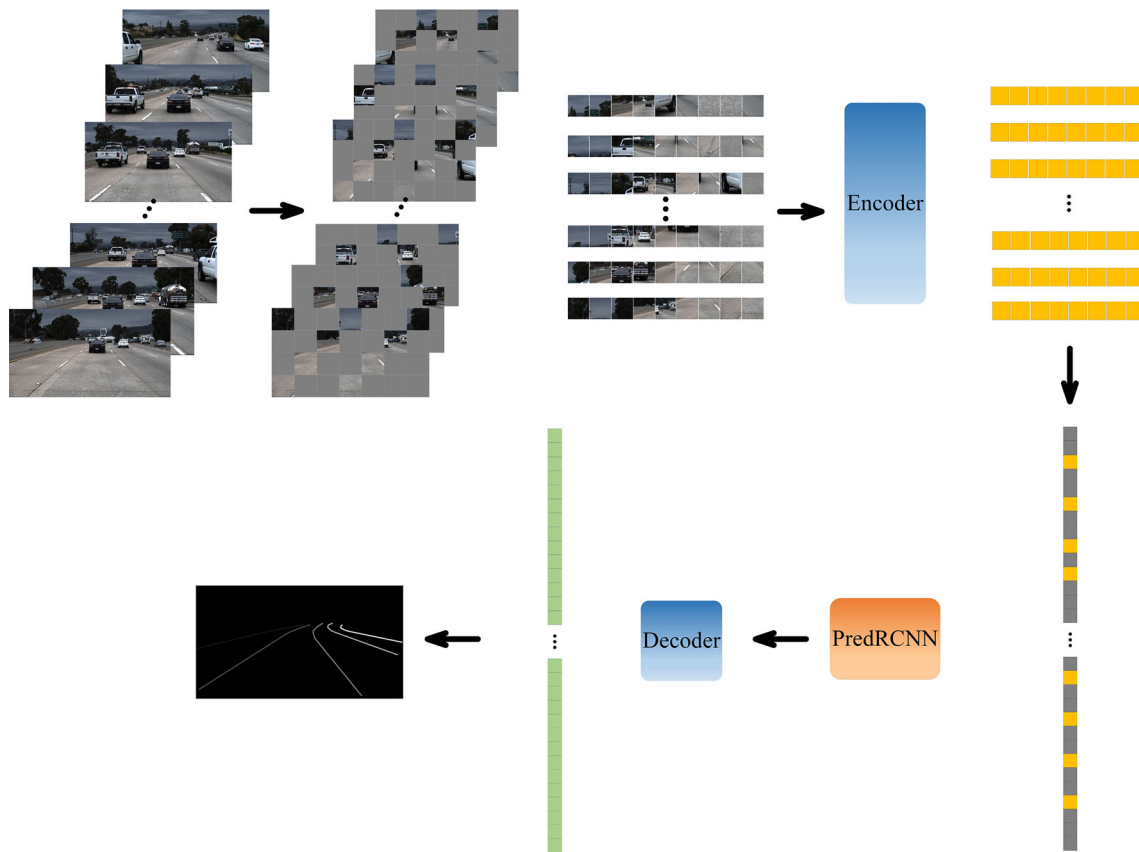
The lane detection network is based on the framework of the masked autoencoders (MAE) and optimized. MAE is a simple autoencoding network that reconstructs the original signal based on some observations of the original signal. Similar to all autoencoding networks, the MAE includes an encoder and a decoder. The function of the encoder is to map the observed signal to the latent variable, and the function of the decoder is to reconstruct the original signal from the latent variable. However, unlike the classical autoencoding network, MAE adopts an asymmetric design, which allows the encoder to operate only on some observed signals (without mask tokens) and uses a lightweight decoder to reconstruct the signal, which utilizes the latent variable and mask tokens.

### Masking

Pictures (size:  $1280 \times 720$ ) are divided into regular nonoverlapping patches. Then, some patches are sampled, and the remaining patches are masked (i.e., removed). Random sampling without replacement and following uniform distribution is called "random sampling". Random sampling with a high masking rate largely eliminates redundancy. Therefore, creating a task that cannot be easily solved by extrapolation from visible neighboring patches and uniform distribution can prevent potential center bias (i.e., more mask patches near the center of the images). Finally, the highly sparse input creates an opportunity to design an efficient encoder, and the effect is shown in Fig. 2.

### Encoder–decoder network based on MAE architecture

Inspired by SegNet, U-Net, and LaneNet [41]'s successful use of encoder–decoder architecture in semantic segmentation and the development of self-attention mechanism Transformer, the Swin Transformer and Masked Auto Encoder have often been used in the field of computer vision in recent years. Based on the MAE framework, this paper proposes an encoder–decoder network by embedding the Swin Transformer block into the encoder–decoder network to replace the original transformer block. To realize lane detection, we build a deep hybrid network based on the encoder–decoder framework to complete the task of semantic segmentation. Because the encoder–decoder network supports the input and output sizes being the same, the whole network can be trained in an end-to-end manner. In the encoder part, Swin Transformer blocks are used for image extraction and feature extraction. In the decoding part, another Swin Transformer block is used to complete the upsampling technique to obtain and highlight



**Fig. 1** Network framework. During pretraining, we first patch each input image. Then, we mask a large quantity of random image block subsets out in each image. A small portion of the visible patch is utilized in the encoder. The mask token is introduced after the encoder. The full set

of encoded patches and mask tokens are processed by a small decoder, which reconstructs the original image in pixels. Finally, the decoder is discarded, and the encoder utilizes the uncorrupted images to produce a representation for the recognition task

the object information and carry out spatial reconstruction to realize lane detection.

**Encoder:** As shown in Fig. 3, we propose an encoder based on a Swin Transformer [41], and the masked image patches (size: “180 × 160”) are input. After patch partitioning, the size of the image patches is “45 × 40 × 48”. Through linear embedding, we set the dimension  $C$  of the vector to 96, and the image size to “45 × 40 × 48”. To capture multiscale features, a hierarchical Swin Transformer block is constructed, and the operation of patch merging as proposed by us. Through a tensor, the adjacent patches are combined into a larger patch to achieve the effect of a downsampling feature map. At this time, the same signals in the window will be merged. Through the above operations, the spatial size is halved, and the number of channels is doubled.

To improve the efficiency of feature extraction and reduce the complexity of calculation, Swin Transformer computes self-attention based on the shift-window. The Swin Transformer block is shown in Fig. 4. Because the computation of global self-attention will lead to the quadratic increase of

complexity, the computation cost will be too high when completing the downstream tasks, especially the dense prediction or high-resolution pictures. The global self-attention computation is replaced by shifted-window computation. Patch the picture evenly into windows without overlap, as shown in the figure below. Patch is the smallest computation unit. There are “ $M \times N$ ” patches in each window.  $M$  and  $N$  are 5 by default. And these 72 windows were computer self-attention. The computation equation of self-attention is as follows:

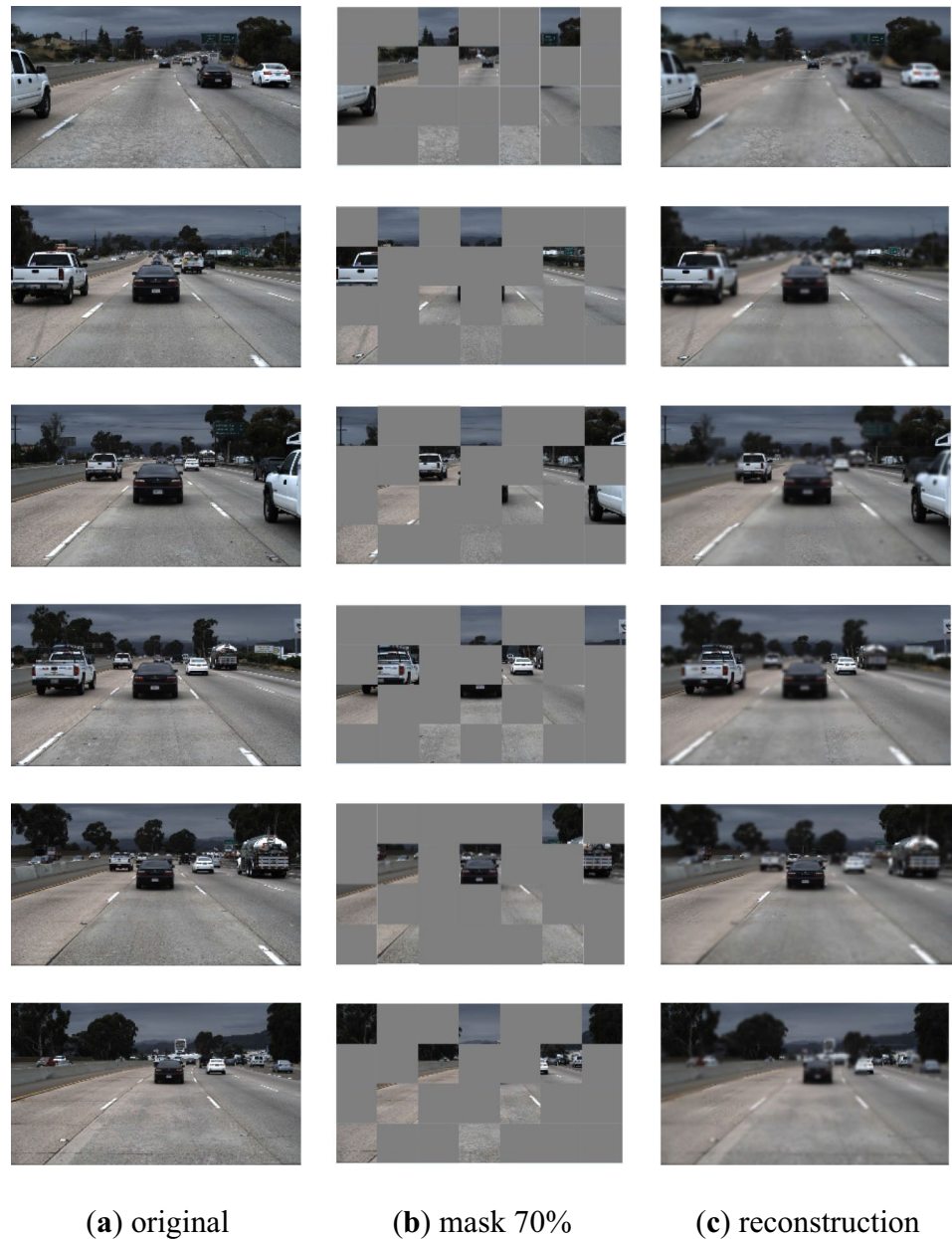
$$\Omega(\text{MSA}) = 4hwC^2 + 2(hw)^2C, \tag{1}$$

$$\Omega(\text{W - MSA}) = 4hwC^2 + 2M N h w C. \tag{2}$$

Equation (1) is the standard self-attention computation, the number of patches is “ $h \times w$ ” for each image, and  $C$  is the feature dimension. Equation (2) is computed based on window self-attention. Through a standard self-attention head, the input is transformed into three matrices:  $q$  (i.e., query),  $k$  (i.e., key) and  $v$  (i.e., value), and the original input vector is multiplied by three coefficient matrices. After the



**Fig. 2** Masking: we show the masked image (a), the ground truth (b), and our optimized MAE reconstruction (c). The masking rate is 70%. When no loss is computed on visible patches, the model output on visible patches is qualitatively worse. This result, however, does not affect the lane detection tasks. It also reduces the cost of calculation



$q$  and  $k$  matrices are obtained, the two matrices are multiplied to obtain the self-attention matrix, and then, the matrix is multiplied by the  $v$  matrix, which is equivalent to weighting. Finally, the original vector dimension is projected to the target dimension through linear projection. This is shown in Fig. 5.

Input vector (i.e., size:  $h \times w \times C$ ) multiplied by three coefficient matrices to obtain  $q$ ,  $k$ , and  $v$  matrices; the complexity of this layer is  $3 \times hw \times C^2$ . Then, the  $q$  and  $k$  matrices are multiplied to obtain the self-attention  $A$  (i.e., attention) matrix with dimensions of  $hw \times hw$ , and the complexity is  $2(hw)^2 \times C$ . When the self-attention matrix  $A$  is multiplied by the  $v$  matrix, the complexity remains

unchanged. Therefore, the complexity of this layer is  $2(hw)^2 \times C$ . The complexity of the last projection linear layer is  $hw \times C^2$ . Therefore, the total complexity is  $4hwC^2 + 2(hw)^2C$ .

Since each window computes the multi-head self-attention, it is directly substituted into Eq. (1). The height and width of the feature map have changed,  $h$  changes  $M$ ,  $w$  changes  $N$ , and Eq. (1) is used to obtain  $4M \times N \times C^2 + 2(MN)^2 \times C$ . The complexity of self-attention is computed in one window, and the number of windows is  $\frac{h}{M} \times \frac{w}{N}$ . Finally, the complexity of computation self-attention in the total window is  $(\frac{h}{M} \times \frac{w}{N}) \times [4M \times N \times C^2 + 2(MN)^2 \times C] = 4hwC^2 + 2MNHwC$ .

It is found that the computational complexity of Eqs. (1) and (2) varies greatly. However, self-attention computation based on windows solves the problem of high computational complexity, but there is no intersection among windows. This results in a lack of global modeling capacity. Therefore, the shifted-window computation not only controls the complexity of computation but also has the capacity of global modeling. The computation equations of the shifted window are as follows:

$$\hat{z}^l = \text{W-MSA}(\text{LN}(z^{l-1})) + z^{l-1}, \tag{3}$$

$$z^l = \text{MLP}(\text{LN}(\hat{z}^l)) + \hat{z}^l, \tag{4}$$

$$\hat{z}^{l+1} = \text{SW-MSA}(\text{LN}(z^l)) + z^l, \tag{5}$$

$$z^{l+1} = \text{MLP}(\text{LN}(\hat{z}^{l+1})) + \hat{z}^{l+1}, \tag{6}$$

where  $\hat{z}^l$  and  $z^l$  represent the output features of (S)WMSA and MLP of block  $l$ , respectively. However, the shifted window has the problem that the size of the window partitioned by shifted-window computation is different from the size of the window partitioned by window computation. If all computation windows are combined into a batch to complete fast computation, self-attention cannot be computed directly. If the windows with different sizes are filled with the same size as the largest window in the middle, the cost and complexity of computation will increase significantly. Therefore, to reduce the computation cost and ensure that the number of computation windows remains unchanged, the computation is carried out in the form of a mask, as shown in Fig. 6.

After partitioning the window, a cyclic shifted window is made to move the position of the original windows 1 and 3 to the position shown in the figure. After cyclic displacement, the position of the original window rotates 180° counter-clockwise. The number of windows remains the same before and after the shift. The computational complexity is also controlled. To solve the problem of computing self-attention between two adjacent windows, self-attention is computed by masking. The computation is completed and restored back to avoid information loss. The visualization of the window mask calculation is shown in Fig. 7. Two lines are drawn in the middle to evenly divide the picture into four windows. The patch in window 0 can compute the self-attention between two. Since the patch parts in Windows 1, 2 and 3 come from different regions, it is impossible to complete the computation of self-attention between two. Therefore, we cover the patches from different regions and then calculate them. After the computation, we operate through SoftMax.

*Decoder* The decoder is used to reconstruct the pixel information of the masked patches. The image patches without masking are converted into latent variables by the encoder.

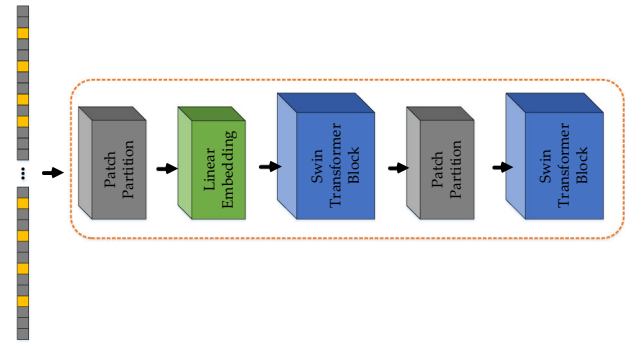


Fig. 3 The structure of the encoder. The masked image patches are input. After patch partitioning. Through linear embedding, we set the dimension  $C$  of the vector

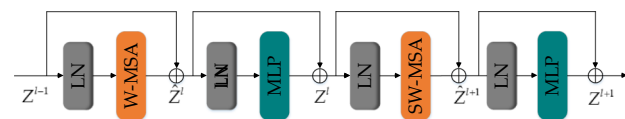


Fig. 4 The structure of the Swin Transformer block

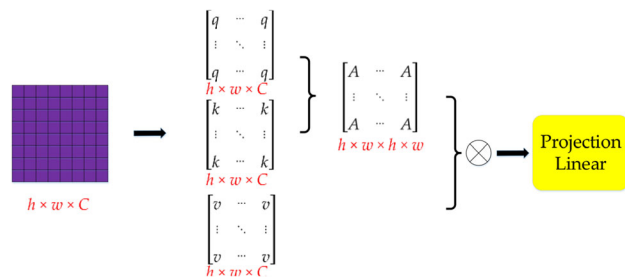


Fig. 5 Standard self-attention computation

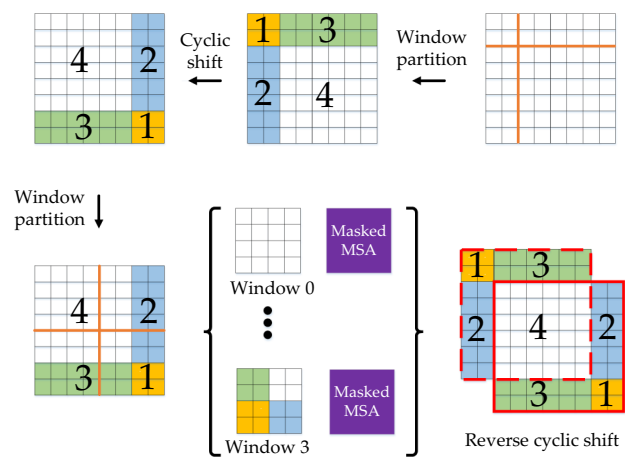
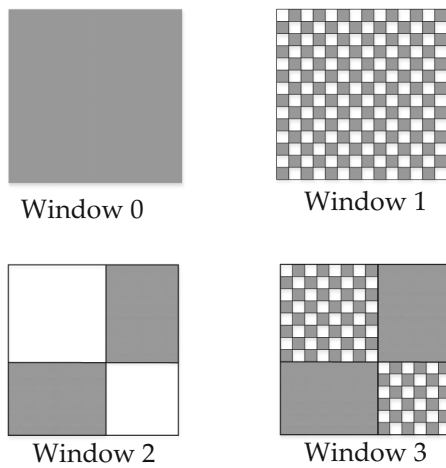


Fig. 6 Masking computation



**Fig. 7** Visualization of window mask computation

All masked patches are represented by a shared vector that can be learned; that is, each masked patch is represented as the same vector, and the value of this vector can be learned. The decoder is another Swin Transformer block, that adds location information to distinguish the corresponding mask. Using a relatively small decoder size, the computational overhead is less than 10% of that of the encoder. The last layer of the decoder is the linear layer. Its function is to project the pixels in a patch to a dimension and then reconstruct the original size to obtain the original pixel information. The loss function adopts the mean squared error (MSE) (i.e., normalization only once) to make the mean value of pixels in each blocked image block to become 0 and the variance to become 1, which is more stable. As a result, the loss function makes the new pixel be different from the original pixel, it sums the squares, and it uses MSE only on the masked patch. Some token columns are generated. The token column is a linear projection of each patch, plus the location information. Next, the random information is scrambled, and the last piece is removed to complete random sampling. When completing the decoding task, one needs to attach mask tokens with the same length as before, which is a vector with learning ability, and add location information. Then, through the unshuffled operation, it is restored to the original order. This is to facilitate one-to-one correspondence with the previous original (i.e., no operation has been done) patch when computing the error.

**PredRNN** The multiple consecutive frames of the driving scene are modeled as time-series. The PredRNN block in the proposal network accepts the feature map extracted by the encoder Swin Transformer on each frame as input. To process various time-series data, different network types of RNN have been proposed, such as LSTM [42] and GRU [43]. We use a spatiotemporal LSTM (ST-LSTM) network, which uses the cells in the network to judge whether the information is

important. The ability to forget unimportant information and remember basic features is usually better than the traditional RNN network. Due to the general computational capabilities of the traditional fully connected LSTM (FC-LSTM), it leads to low efficiency. This paper uses double-layer ST-LSTM in the proposed network, which is widely used in end-to-end training and feature extraction of time-series. The activation of the ST-LSTM cell at time  $t$  can be expressed as

$$\begin{aligned}
 g_t &= \tanh(W_{xg} \times X_t + W_{hg} \times H_{t-1}^l + b_g) \\
 i_t &= \sigma(W_{xi} \times X_t + W_{hi} \times H_{t-1}^l + b_i) \\
 f_t &= \sigma(W_{xf} \times X_t + W_{hf} \times H_{t-1}^l + b_f) \\
 C_t^l &= f_t \odot C_{t-1}^l + i_t \odot g_t \\
 g_t' &= \tanh(W'_{xg} \times X_t + W_{mg} \times M_t^{l-1} + b'_g) \\
 i_t' &= \sigma(W'_{xi} \times X_t + W_{mi} \times M_t^{l-1} + b'_i) \\
 f_t' &= \sigma(W'_{xf} \times X_t + W_{mf} \times M_t^{l-1} + b'_f) \\
 M_t^l &= f_t' \odot M_t^{l-1} + i_t' \odot g_t' \\
 o_t &= \sigma(W_{xo} \times X_t + W_{ho} \times H_{t-1}^l + W_{co} \times M_t^l + b_o) \\
 H_t^l &= o_t \odot \tanh(W_{1 \times 1} \times [C_t^l, M_t^l]).
 \end{aligned} \tag{7}$$

Both  $C$  and  $M$  units are reserved, and  $C_t$  is the standard time unit, which is propagated from the previous node  $t - 1$  to the current time step in each ST-LSTM unit. The  $M_t$  of layer  $L$  is the spatiotemporal memory described, which is vertically propagated from layer  $L - 1$  to the current node in the same time step.  $M$  of the last layer at time  $t - 1$  will be propagated to the first layer at time  $t$ . Another group of door structures is constructed for  $M_t$  while retaining the original door control mechanism of  $C_t$ . Finally, the final hidden state of the node depends on the fused spatiotemporal memory. These memories from different directions are connected and  $1 \times 1$  is applied. The dimension of the convolution layer is reduced, so that the hidden state  $H_t$  has the same dimension as the memory cell. Different from simple memory splicing, the ST-LSTM unit uses a shared output gate for the two memory types to achieve seamless memory fusion, which can effectively model the shape deformation and motion trajectory in the spatiotemporal sequence. In the network, the input and output sizes of the ST-LSTM are equal to the size of the feature map generated by the encoder.

### Training strategy

To achieve the effect of accurate prediction, an end-to-end trainable deep hybrid neural network is established, and the network is trained through the backpropagation process. In



the process of backpropagation, the weight parameters of the deep hybrid network are updated.

First, the proposed network is pretrained on ImageNet [44], and the pretraining weight is used for initialization. This not only saves training time but also propagates the appropriate weight to the proposed network. The test accuracy of training without pretraining weight initialization is very close to that of training with pretraining weight initialization, but the former takes more time. In this paper, the input of the network is  $N$  continuous images of various driving scenes. Therefore, in backpropagation, the coefficient of each weight update of ST-LSTM should be divided by  $N$ . In our experiment, we set  $N = 6$ .

Second, a loss function is constructed based on weighted cross-entropy to solve the discriminative segmentation task with the following:

$$\varepsilon_{loss} = \sum_{X \in \Omega} \omega(X) \log(p_{l(X)}(X)), \tag{8}$$

where  $l: \Omega \rightarrow \{1, \dots, K\}$  is the real label of each pixel, and  $w: \Omega \rightarrow R$  is the weight of each level, which is used to balance the lane level and is set as the ratio of the number of pixels in the two categories in the training set. SoftMax is defined as follows:

$$p_k(X) = \frac{\exp(a_k(X))}{\sum_{k'=1}^K \exp(a_{k'}(X))}, \tag{9}$$

where  $a_k(X)$  represents the activation in characteristic channel  $k$  at the pixel position.  $\chi \in \Omega, \Omega \in Z^2$ , where  $k$  is the number of classes. We use different optimizers in different training stages to make the training network better. First, the adaptive motion estimation (Adam) optimizer [45] is adopted to make the gradient descent rate faster. Because it is easy to fall into local minima and difficult to converge, when the network is trained to a relatively high accuracy, we turn to the stochastic gradient descent (SGD) [46] optimizer and use the fine step size to find the global optimum. When changing the optimizer, to avoid the confusion or stagnation of the convergence process caused by the interference of different learning steps during training, it is necessary to match the learning rate. The matching equation of the learning rate is as follows:

$$\begin{aligned} \omega_{k_{sgd}} &= \omega_{k_{Adam}}, \\ \omega_{k-1_{sgd}} &= \omega_{k-1_{Adam}}, \\ \omega_{k_{sgd}} &= \omega_{k-1_{sgd}} - \alpha_{k-1_{sgd}} \hat{\nabla} f(\omega_{k-1_{sgd}}), \end{aligned} \tag{10}$$

where  $\omega_k$  represents the weights in the  $k$ th iteration,  $\alpha_k$  represents the learning rate, and  $\hat{\nabla} f(\cdot)$  is a random gradient calculated by the loss function  $f(\cdot)$ . In the experiments, the

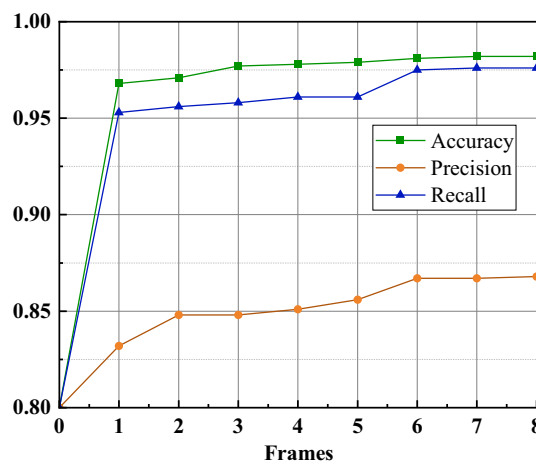


Fig. 8 Test of whether the number of frames in the input image affects network performance based on our designed network. Among them, we select accuracy, precision, and recall as evaluation indicators. The number of frames increases from 1 to 8

initial learning rate is set to 0.01, and the optimizer is changed when the training accuracy reaches 90%.

### Experiment and results

In this section, we report the experiments that were carried out to verify the accuracy and robustness of the proposed network. The performance of the proposed network was evaluated in different scenarios and compared with different lane detection networks, and the influence of parameters was analyzed. This section is divided into four parts: (1) Datasets; (2) Hyperparameter Settings and Hardware Environment; (3) Performance Evaluation; (4) Visualization.

#### Datasets

A dataset is constructed based on the TuSimple [47] lane dataset. The TuSimple Dataset is divided into a training set and a test set. The training set contains 3626 image sequences (i.e., the front view of the driving scene on the highway) with each sequence containing 20 continuous frames. The test set contains 2944 image sequences with each sequence containing 20 continuous frames. We label an additional 10 frames of images in each sequence to expand the dataset. We study the frame number of the input picture, and select the different frame number of the picture frame number input to our network. The experimental results are shown in Fig. 8. When the number of frames in the input picture increases from 1 to 6, the performance of the network is improved significantly. When the frame number of input picture is more than 6, the improvement of network performance is limited. Therefore, considering the equipment performance and the

**Table 1** Sampling methods of continuous input images

Stride	Sampled frames	Ground truth
1	11th 12th 13th 14th 15th 16th	16th
2	6th 8th 10th 12th 14th 16th	16th
3	1th 4th 7th 10th 13th 16th	16th
1	13th 14th 15th 16th 17th 18th	18th
2	8th 10th 12th 14th 16th 18th	18th
3	3th 6th 9th 12th 15th 18th	18th

computational cost, we choose 6 consecutive frames as the input.

In the training process, 6 consecutive frames and the ground truth value of the last frame are used as inputs, and the lane is detected in the last frame. The training set is constructed based on the ground truth labels in the 16th and 18th frames. At the same time, to make the proposed network fully adapt to lane detection at different driving speeds, we sample the input image with three different steps (i.e., at intervals of 1, 2, and 3 frames). There are three sampling methods for real labels on the ground, as shown in Table 1.

In the test, we use 6 continuous images to identify the lane in the last frame and compare it with the real situation of the ground in the last frame. Testset#1 and Testset#2 are constructed. Testset #1 is built based on the TuSimple test set and used for the normality test. Testset #2 consists of samples collected under different vehicle conditions and is used to evaluate the robustness of network detection. The dataset settings are shown in Table 2. To further evaluate the robustness of network detection, we also built Testset#1 and Testset#2 based on CULane. Testset#1 is used for normality test, and Testset#2 is used to evaluate the robustness of network detection. The dataset settings are shown in Table 3.

## Hyperparameter settings and hardware environment

In the experiments, the sampling resolution of the lane detection image is adjusted to  $368 \times 640$  to save memory, make up for some blurred lane boundaries and to protect the network from the complex texture in the background around the vanishing point. To verify the applicability of low-resolution images and the performance of the proposed lane detection network, different test conditions are used for wet, the cloudy and sunny scenes. In the training phase, the batch size is 32, the number of epochs is 8000, the activation function is LReLU, and the network framework is Pytorch 1.7.1. The running time is recorded using a GPU (RTX3080 GPU), and the final value of the running time is obtained after averaging the running time of 1000 samples. All experiments were

completed on the i7-11700k platform equipped with a dual NVIDIA RTX3080 GPU and 11th Gen Intel Core i7-11700K.

## Performance evaluation and comparison

In this section, we describe two main experiments. First, the performance of the network is evaluated quantitatively. Second, the robustness of the network is verified based on the effect of completing the test task in various scenarios.

*Overall performance* The proposed networks, i.e., UNet-ST-LSTM, SegNet-ST-LSTM, LaneNet-ST-LSTM, and ST-MAE-ST-LSTM, are compared with their original baselines, as well as some modified versions. To be specific, the following methods are included in the comparison.

*SCNN* Replace traditional deep layer-by-layer convolutions with slice-by-slice convolutions. Segment with backbone + SCNN structure, LargeFOV (Deeplabv2) is selected for backbone.

*ResNet 18, ResNet 34* The residual network is composed of a series of residual blocks. The input is convoluted many times and then added to the input. 18 and 34 represent the number of weight layers.

*ENet* The network structure of ENet refers to ResNet, and its structure is described as a main branch and an additional branch with convolution kernel. Finally, the pixel level of addition and fusion is carried out.

*SegNet* It is a classical encoder–decoder architecture neural network for semantic segmentation. The encoder is the same as VGGNet.

*SegNet-ST-LSTM* It is a hybrid neural network we proposed, using spatiotemporal LSTM (ST-LSTM) after the encoder network.

*Six UNet-based networks, LaneNet-based networks, and ST-MAE-based networks* Replacing the encoder and decoder of SegNet with that of a modified U-Net, LaneNet and ST-MAE, as introduced in “Method”, it generates another six networks accordingly.

After completing the training of the above deep hybrid network, the results obtained are compared on the test set. Through quantitative comparison, it is shown that the proposed framework is excellent and that the network can correctly detect the number of lanes in the image sequence. In the lane detection task, two kinds of detection errors should be avoided (the number of detected lanes is inconsistent with the real number of lanes). One is missed detection. The real lane object is used as the background of the predicted image sequence, so the number of lanes cannot be detected correctly. The second is over-detection, which detects other objects in the image sequence as lanes. When the network

**Table 2** Structure and content of the original dataset based on TuSimple

Part of dataset	Including	Resolution	#Lane	Environment	Labeled frames	Labeled images
Train	TuSimple	1280 × 720	≤ 4	Highway	16th and 18th	7252
Test	Testset #1	1280 × 720	≤ 4	Highway	16th and 18th	2944
	Testset #2	1280 × 720	≤ 4	Highway	All frames	598

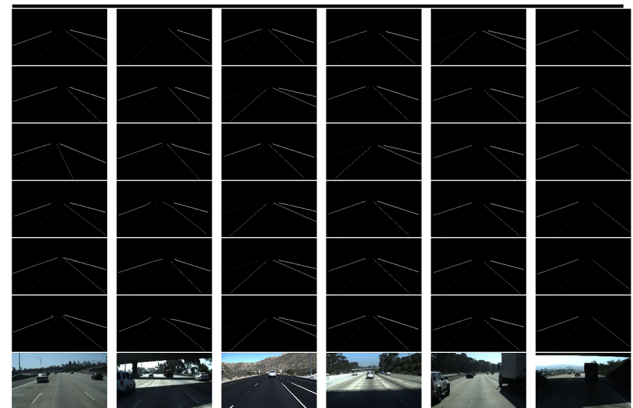
**Table 3** Structure and content of the original dataset based on CULane

Part of dataset	Including	Resolution	#Lane	Environment	Labeled frames	Labeled images
Train	CULane	1640 × 590	≤ 4	City, rural, highway	16th and 18th	9458
Test	Testset #1	1640 × 590	≤ 4	City, rural, highway	16th and 18th	3840
	Testset #2	1640 × 590	≤ 4	City, rural, highway	All frames	780

meets the above conditions, the problems of serious disconnection and fuzzy areas in the detection diagram and the large difference in position and length between the lane and the ground should be avoided. The experimental results show that the network achieves the above two objectives, as shown in Figs. 8, 10, and the comparison detection with other networks is shown in Figs. 9, 11. The network can detect each lane in the input image sequence in different scenes (including complete lane marking lines, shaded lane marking lines, and lane marking lines blocked by other objects), without missing detection and over-detection. Through the comparative experiment, the position of the lane detected by the network designed in this paper is consistent with the real lane position, and lane detection by other networks is incomplete or at a certain distance from the real lane position. By comparing the lane detection results in this paper with others, the lane detection results in this paper appear as thin white lines, with fewer fuzzy areas, and are not in the area of severe shadow occlusion and the fuzzy prediction caused by driving vehicle occlusion. In scenes involving incomplete road lane marks, our network can complete the task of lane detection.

In addition, to accurately evaluate the network performance, we use quantitative analysis to test the superiority of network performance. Accuracy, Precision, and Recall are used as evaluation indices to make a fair and reasonable comparison. In addition, the false-positive, false-negative, true-positive, true-negative, true-positive rate and false-positive rate are also reported. Among them, the TPR, FPR, accuracy, precision, and recall calculation methods, such as Eqs. (11), (12), (13), (14), and (15), show the following:

$$TPR = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}, \tag{11}$$



**Fig. 9** Visual comparison of network lane detection results in this paper. Line 7: ground truth. The rest: lane semantic segmentation diagram

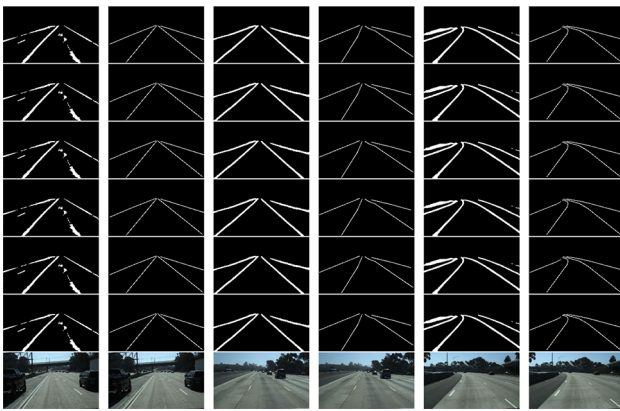
$$FPR = \frac{\text{False Positive}}{\text{False Positive} + \text{True Negative}}, \tag{12}$$

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{Total Number of Pixels}}, \tag{13}$$

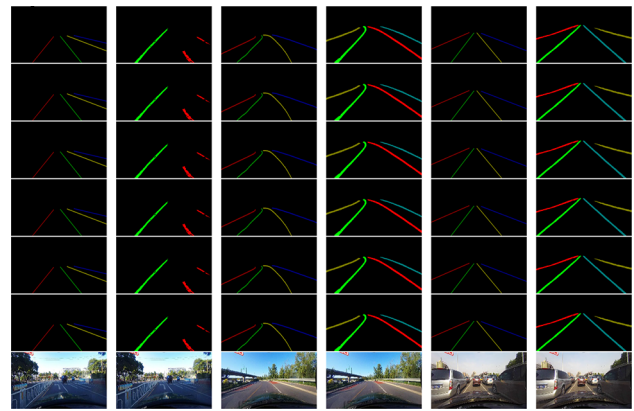
$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}, \tag{14}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}. \tag{15}$$

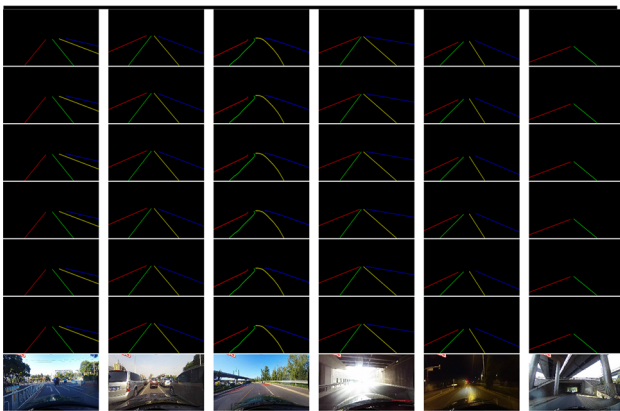
In the lane detection task, the negative class and positive class are defined as the background and lane, respectively. True positives are defined as the number of lane pixels correctly predicted, false-positive positives are defined as the number of background pixels incorrectly predicted, and false negatives are defined as the number of lane pixels incorrectly



**Fig. 10** Comparison of lane detection tasks completed by this network and other networks in the same scenario based on TuSimple dataset. Column 1 and Column 2: network comparison diagram between ENet and our network; Columns 3 and 4: comparison diagram between LaneNet and our network; Columns 5 and 6: comparison diagram between ResNet-34 and our network



**Fig. 12** Comparison of lane detection tasks completed by this network and other networks in the same scenario based on CULane dataset. Column 1 and Column 2: network comparison diagram between our network and ENet; Columns 3 and 4: comparison diagram between our network and LaneNet; Columns 5 and 6: comparison diagram between our network and ResNet-34



**Fig. 11** Performance of ST-MAE based on the CULane test set. Six most challenging scenarios were selected for test. Among them, the first column: Arrow, the second column: Crowded, the third column: Curve, the fourth column: Dazzle light, the fifth column: Night, and the sixth column: No line. The last row of the figure denotes inputs, and the first row to the sixth row, respectively, denotes instance segmentations after clustering from the selected six consecutive image sequences

predicted as the background. After adding ST-LSTM, the accuracy is improved, and the recall rate is approximately the best. Compared with the original version, the accuracy of ST-LSTM is improved by 7%, and the recall rate is only reduced by 3%. After adding ST-LSTM to SegNet, the accuracy is improved by 6%, and the recall rate is slightly improved.

By observing the experimental results in Figs. 9 and 10, it can be concluded that the reduction of the fuzzy adhesion area, high robustness of detection in various scenes and more detailed detection results are the main reasons for the high accuracy of the network designed in this paper. The network mainly reduces the possibility of misclassifying the background pixels close to the ground into lanes and reduces the

fuzzy area around the vanishing point and vehicle occlusion area to reduce the false alarm rate. Figures 11, and 12 show that after using the network in this paper to complete the lane detection task, the problem of other boundaries being incorrectly detected as lanes by ENet [48] and ResNet-34 [46] is improved.

Because all pixels in an area predicted by the network are lane categories, this will form adhesions, and all lane pixels will be correctly classified, leading to a high recall rate. In this case, even if the recall is very high, there will be serious misclassification of background classes, and the accuracy is very low. Considering that the accuracy or recall rate only reflects one aspect of lane detection performance, the evaluation is not comprehensive. Therefore, we introduce the F1 measure as an evaluation index of the overall matrix for evaluation. F1 is calculated as follows:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (16)$$

In Eq. (16), the weight of Accuracy is the same as that of recall. As shown in Tables 4, 5, 6 and 7 compared with the original version of the network in this paper, the measured value of network F1 with added ST-LSTM increases by approximately 2%. The input multi-frame image sequence can better predict the lane than the input single-frame image sequence, and ST-LSTM can better process continuous sequence images in the semantic segmentation framework. Therefore, the introduction of ST-LSTM can accept high-dimensional tensors as input and is improved based on the original baseline.

To evaluate the performance of this network, based on the TuSimple dataset and CULane dataset, we compare this network with other networks through experiments further



**Table 4** Comparison of true-positive rate and false-positive rate based on TuSimple dataset

Algorithm	TPR		FPR	
	Easy lanes	Hard lanes	Easy lanes	Hard lanes
ResNet-18	0.921	0.912	0.078	0.082
ResNet-34	0.934	0.925	0.064	0.069
SCNN	0.945	0.936	0.053	0.058
ENet	0.947	0.938	0.049	0.054
UNet	0.953	0.945	0.042	0.046
SegNet	0.956	0.947	0.039	0.045
LaneNet	0.962	0.949	0.034	0.041
ST-MAE	0.967	0.954	0.031	0.035
UNet_ST-LSTM	0.964	0.953	0.032	0.033
SegNet_ST-LSTM	0.968	0.957	0.031	0.032
LaneNet_ST-LSTM	0.972	0.961	0.027	0.029
ST-MAE_ST-LSTM	0.975	0.971	0.024	0.025

**Table 5** Comparison of true-positive rate and false-positive rate based on CULane dataset

Algorithm	TPR		FPR	
	Easy lanes	Hard lanes	Easy lanes	Hard lanes
ResNet-18	0.902	0.897	0.102	0.115
ResNet-34	0.913	0.908	0.084	0.078
SCNN	0.929	0.914	0.072	0.063
ENet	0.938	0.918	0.061	0.057
UNet	0.941	0.929	0.052	0.059
SegNet	0.944	0.931	0.047	0.056
LaneNet	0.957	0.942	0.034	0.041
ST-MAE	0.961	0.949	0.033	0.037
UNet_ST-LSTM	0.949	0.936	0.049	0.052
SegNet_ST-LSTM	0.948	0.939	0.044	0.049
LaneNet_ST-LSTM	0.962	0.948	0.032	0.038
ST-MAE_ST-LSTM	0.964	0.961	0.029	0.031

objectively in Tables 8 and 9. Different from the pixel-level test standard used, the prediction points are sparsely sampled. Because the clipping and resizing operations are used in the preprocessing step of building the dataset, the prediction points obtained by sparse sampling should be consistent with the corresponding points in the original image.

**Running time** With the image sequence as the input and the addition of ST-LSTM, the running time may be greater. As seen from the last column of Table 6, processing images of all six frames shows more time consumption than processing images of only one frame. However, in this paper, the network can complete the lane detection task online. Because the previous frame has been extracted, the encoder only needs to process the current frame, and hence, the running time will not increase. ST-LSTM can run in parallel mode on the GPU,

and the running time is almost the same. After adding the ST-LSTM block in this network, if all six frames are processed as new input, the running time is approximately 21 ms. If the first five frames are stored and reused, the running time is 4.4 ms which is slightly longer than the original network running time of 3.1 ms.

**Robustness** We have achieved good accuracy in the test set, but we still need to test the robustness of the lane detection network; that is, whether the lane detection network can complete the lane detection task well in various scenes and vehicle conditions. In the robustness test experiment, new datasets are used, including various driving scenes, such as daily driving scenes involving urban roads and highways, as well as challenging driving scenes such as incomplete or even no lane identification, poor line of sight, and vehicle

**Table 6** Comparison of recall, F1-measure, and running time based on TuSimple dataset

Algorithm	Recall	F1-measure	Running time
ResNet-18	0.921	0.792	0.0078
ResNet-34	0.934	0.803	0.0064
SCNN	0.945	0.816	0.0053
ENet	0.947	0.854	0.0049
UNet	0.953	0.867	0.0053
SegNet	0.956	0.861	0.0046
LaneNet	0.962	0.884	0.0042
ST-MAE	0.967	0.893	<b>0.0031</b>
UNet_ST-LSTM	0.964	0.897	0.0065
SegNet_ST-LSTM	0.968	0.901	0.0069
LaneNet_ST-LSTM	0.972	0.914	0.0055
ST-MAE_ST-LSTM	<b>0.975</b>	<b>0.919</b>	0.0044

The best performance of Recall, Running time and F1-Measure of each networks in the experiment are shown in bold

**Table 7** The performance of different algorithms on the CULane test set

Category	ResNet-18	ResNet-34	UNet	LaneNet	SCNN	ST-MAE	UNet_ST-LSTM	SegNet_ST-LSTM	LaneNet_ST-LSTM	ST-MAE_ST-LSTM
Normal	90.5	90.2	91.1	91.7	90.6	94.2	92.3	93.1	93.6	95.3
Crowded	65.6	67.5	68.2	70.3	69.7	75.1	69.1	70.4	72.2	78.9
Dazzle light	67.7	60.1	61.2	60.3	58.5	71.5	61.9	62.3	62.7	73.2
Shadow	65.9	68.2	67.1	67.5	66.9	79.2	68.3	68.9	69.4	80.4
No line	40.7	42.5	43.3	44.6	43.4	54.3	45.1	45.7	46.7	57.4
Arrow	83.7	84.3	85.4	85.2	84.1	86.1	86.7	87.2	87.9	89.2
Curve	61.2	62.4	63.7	65.1	64.4	66.7	64.5	65.2	66.8	68.1
Crossroad	1848	1981	2044	2005	1990	1901	2051	2018	2013	1952
Night	64.5	65.7	67.3	67.4	66.1	71.8	68.1	69.2	70.3	73.2
Total	70.7	71.2	72.3	73.7	71.3	77.9	74.2	75.4	78.7	83.3
Frames per second (fps)	25.8	37.1	50.1	48.8	8.2	64.8	49.3	46.9	47.3	64.2

blocking lanes. Experiments show that Validation–Accuracy is 97.46%, Test–Accuracy is 97.37%, and Precision is 0.865 in simple scenes. Validation–Accuracy is 97.38%, Test–Accuracy is 97.29%, and Precision is 0.859 in complex scenes. Therefore, the detection network proposed in this paper has strong robustness.

F1 values are shown for different categories of scenarios in Table 7. FP values are shown for the scenario of crossroad. As there is no straight line at the scenario of crossroad, any prediction point is a false positive.

### Ablation study

We investigate the effects of the model with only ST-MAE (shown in Table 10) and perform extensive experiments to

investigate the effects of different locations of double-layers-ST-LSTM (shown in Table 11), e.g., embedding double-layers-ST-LSTM into before encoder or after decoder.

The performance of the model with only ST-MAE, LaneNet, and embedding double-layers-ST-LSTM into ST-MAE, LaneNet is discussed, which to demonstrate that the double-layer ST-LSTM has the positive improvement on the continuous multi-frame lane detection network. The experimental results show the following: (1) the performance of same time interval is better than the different time interval. The reason may be that equal interval between frames makes the missing information present regularity and stability, which enables double-layers-ST-LSTM to look back in the past better and predict the future frame well. However, the sequential and unequal interval between frames

**Table 8** Comparison of accuracy and precision based on TuSimple dataset

Algorithm	Val_Acc (%)		Test_Acc (%)		Precision	
	Easy lanes	Hard lanes	Easy lanes	Hard lanes	Easy lanes	Hard lanes
ResNet-18	92.85	91.42	92.68	91.45	0.671	0.659
ResNet-34	93.15	92.63	93.22	92.58	0.695	0.672
SCNN	94.12	92.85	94.25	93.12	0.712	0.695
ENet	94.98	93.62	94.73	93.74	0.724	0.708
UNet	95.42	94.14	95.53	94.08	0.763	0.749
SegNet	95.76	94.57	95.61	94.52	0.784	0.765
LaneNet	96.56	95.35	96.48	95.42	0.812	0.803
ST-MAE	96.82	96.69	96.85	96.72	0.843	0.835
UNet_ST-LSTM	96.38	95.95	96.19	95.82	0.851	0.847
SegNet_ST-LSTM	96.51	96.12	96.43	95.98	0.863	0.854
LaneNet_ST-LSTM	97.04	96.45	96.94	96.46	0.872	0.865
ST-MAE_ST-LSTM	97.46	97.38	97.37	97.29	0.865	0.859

**Table 9** Comparison of accuracy and precision based on CULane dataset

Algorithm	Val_Acc (%)		Test_Acc (%)		Precision	
	Easy lanes	Hard lanes	Easy lanes	Hard lanes	Easy lanes	Hard lanes
ResNet-18	92.15	90.38	91.92	90.27	0.612	0.598
ResNet-34	92.56	91.58	92.42	90.26	0.618	0.602
SCNN	93.03	91.74	92.97	91.63	0.626	0.612
ENet	93.72	92.21	93.54	91.98	0.659	0.643
UNet	94.21	93.03	93.89	92.74	0.686	0.648
SegNet	94.54	93.42	94.13	93.51	0.702	0.681
LaneNet	95.69	94.57	95.08	94.64	0.753	0.742
ST-MAE	96.17	95.85	95.72	95.41	0.819	0.795
UNet_ST-LSTM	94.87	93.89	94.51	93.27	0.715	0.694
SegNet_ST-LSTM	95.04	94.32	94.83	93.79	0.729	0.741
LaneNet_ST-LSTM	96.37	95.05	95.41	95.84	0.824	0.809
ST-MAE_ST-LSTM	97.03	96.18	96.84	95.92	0.837	0.829

**Table 10** Performance comparison of different module in network on TuSimple dataset and CULane dataset

Location	TuSimple dataset				CULane dataset			
	Acc (%)	Pre	Rec	F1-M	Acc (%)	Pre	Rec	F1-M
ST-MAE	95.75	0.835	0.967	0.893	94.67	0.795	0.765	0.779
LaneNet	95.42	0.812	0.962	0.881	94.64	0.753	0.721	0.737
LaneNet_ST-LSTM	96.46	0.865	0.965	0.912	95.84	0.824	0.753	0.787
ST-MAE_ST-LSTM	97.29	0.859	0.975	0.913	95.92	0.819	0.847	0.833

**Table 11** Performance comparison of different locations of double ST-LSTM in ST-MAE on TuSimple dataset and CULane dataset

Location	TuSimple dataset				CULane dataset			
	Acc (%)	Pre	Rec	F1-M	Acc (%)	Pre	Rec	F1-M
Encoder (front)	91.35	0.756	0.812	0.783	85.64	0.718	0.795	0.755
Encoder (behind)	95.42	0.828	0.895	0.860	89.87	0.794	0.813	0.803
Decoder (front)	97.29	0.859	0.975	0.913	95.92	0.819	0.847	0.833
Decoder (behind)	96.84	0.835	0.943	0.886	94.97	0.759	0.814	0.786

fluctuates greatly and destroys the regularity and stability of information. (2) Under the same conditions, the performance of the model with embedding double-layers-ST-LSTM into ST-MAE is better than that of the model with only ST-MAE. The possible reason is that double-layers-ST-LSTM can try its best to memorize and retain the most likely features of lane.

We have discussed the experimental results of different locations of the double-layers-ST-LSTM in our proposed model, as shown in Table 10. The experimental results show the following: (1) When trying to embed the double-layers-ST-LSTM into the lowest layer (front encoder), the performance of the corresponding model is not ideal. The possible reason is that the lowest layer contains local information. (2) When trying to embed the double-layers-ST-LSTM into the highest layer (behind decoder), the performance of the corresponding model is not ideal. The possible reason is that the highest layer mainly contains global information. (3) When trying to embed the double-layers-ST-LSTM close to a middle-level layer (behind encoder), the performance of the corresponding model is also not ideal. The possible reason is that the double-layers-ST-LSTM contains limited global information that not conducive to obtaining location information. (4) When trying to embed the double-layers-ST-LSTM close to a middle-level layer (front decoder), the performance of the corresponding model is ideal. The possible reason is that the double-layers-ST-LSTM may act as a connector between the local and global information. Therefore, we set the double-layers-ST-LSTM at a layer (front decoder) in our work.

## Discussion

*Parameter analysis* The number of frames of the input image sequence and the sampling step size will affect the performance of the proposed network. When inputting a multi-frame image sequence, more information can be obtained in the prediction graph, which is helpful for the final detection result. However, using too many previous frames may lead to poor final detection results. In particular, the scene information in the previous frame that is far away from

the current frame may differ greatly from the current frame and this will have a negative influence on the detection results. Therefore, it is necessary to analyze the impact of the number of input image sequences on lane detection. First, the number of input image sequences is set to range from 1 to 6, and the results obtained are compared with the sampling step size. The test is conducted on test set #1, and the test results are shown in Tables 12, 13. In the top row of Tables 12, 13 the two numbers in each parenthesis are the sampling step size and the number of image frames (i.e., the number of image sequences). When inputting more consecutive frame images with the same sampling step, the accuracy and F1 measurement value are improved. Compared with inputting only the current frame image, multiple consecutive frame images will be more conducive to completing the lane detection task. When the step size increases, the test performance tends to be stable. The test results of inputting images from the fifth frame to the sixth frame are not as good as those of inputting images from the second frame to the third frame. This may be because the information from the farther previous frame is not as helpful for lane prediction and detection as the information from the closer previous frame. When we focus on the sampling step size, as shown in Tables 12, 13, when the number of frames of the input image sequence remains unchanged, the test results of our network under different sampling steps show little difference. In summary, when inputting a multi-frame image sequence, ST-LSTM integrates the feature mapping extracted from the continuous image sequence, which can obtain more complete lane information and better complete the lane detection task.

## Conclusions

In this paper, the masked autoencoders (MAE), Swin Transformer, and PredRNN are designed together, and a deep hybrid network structure that can complete the task of lane detection in various driving scenes is proposed. The network is based on the MAE framework, takes multiple continuous frame image sequences as input, and completes the task of lane detection by semantic segmentation. We input the continuous six-frame image sequence into the network, in which



**Table 12** Results of different hyperparameter settings based on CULane dataset

(Stride, frames)	(3, 6)	(3, 5)	(3, 4)	(3, 3)	(3, 2)	(2, 6)	(2, 5)	(2, 4)	(2, 3)	(2, 2)	(1, 6)	(1, 5)	(1, 4)	(1, 3)	(1, 2)	Single
Total range	15	12	9	6	3	10	8	6	4	2	5	4	3	2	1	–
Accuracy	0.970	0.965	0.966	0.960	0.957	0.970	0.968	0.965	0.961	0.956	0.969	0.968	0.966	0.958	0.957	0.943
Precision	0.854	0.845	0.849	0.837	0.821	0.856	0.845	0.839	0.837	0.821	0.856	0.844	0.838	0.837	0.821	0.804
Recall	0.964	0.949	0.958	0.956	0.953	0.964	0.951	0.946	0.944	0.942	0.964	0.949	0.947	0.945	0.942	0.936
F1-measure	0.908	0.886	0.883	0.878	0.876	0.908	0.886	0.883	0.877	0.875	0.904	0.886	0.882	0.875	0.876	0.842

**Table 13** Results of different hyperparameter settings based on TuSimple dataset

(Stride, frames)	(3, 6)	(3, 5)	(3, 4)	(3, 3)	(3, 2)	(2, 6)	(2, 5)	(2, 4)	(2, 3)	(2, 2)	(1, 6)	(1, 5)	(1, 4)	(1, 3)	(1, 2)	Single
Total range	15	12	9	6	3	10	8	6	4	2	5	4	3	2	1	–
Accuracy	0.981	0.979	0.977	0.971	0.968	0.981	0.979	0.977	0.971	0.968	0.981	0.979	0.977	0.971	0.968	0.954
Precision	0.867	0.856	0.851	0.848	0.832	0.867	0.856	0.851	0.848	0.832	0.867	0.856	0.851	0.848	0.832	0.815
Recall	0.975	0.961	0.958	0.956	0.953	0.975	0.961	0.958	0.956	0.953	0.975	0.961	0.958	0.956	0.953	0.947
F1-measure	0.919	0.897	0.894	0.889	0.887	0.919	0.897	0.894	0.889	0.887	0.919	0.897	0.894	0.889	0.887	0.851

the scene information of each frame image is extracted by an encoder composed of a Swin Transformer block and then input into PredRNN. Multiple continuous frames of the driving scene are modeled as time-series by the ST-LSTM block to effectively model the shape deformation and motion trajectory in the spatiotemporal sequence. Finally, through the decoder composed of Swin Transformer blocks, the features are obtained and reconstructed to complete the detection task. Many experiments were carried out based on the Tusimple dataset to complete the performance evaluation. The results show that compared with the baseline architecture using a single-frame image sequence as input, this architecture achieves better results. The results also verify the effectiveness of using multiple continuous frame image sequences as input. Compared with other networks, our proposed network has higher accuracy, recall, and excellent detection performance. In addition, tests were also carried out in the face of challenging driving scenes to check its robustness. The test results show that the network can stably complete the task of lane detection in various scenarios. Finally, in the parameter analysis, it is found that the input multi-frame image sequence can improve the lane detection results, which further proves that the continuous multi-frame image sequence is more conducive to the completion of the lane detection task than the single-frame image sequence.

Currently, the network training strategy based on deep learning mostly adopts supervised learning. In the future, we plan to train the network through self-supervised learning. It simplifies the network training process, improves the training performance, and has higher robustness in the face of various tasks.

**Funding** This project is supported by the National Natural Science Foundation of China (Grant No. 51605003 and 51575001), the Natural Science Foundation of the Anhui Higher Education Institutions of China under Grant No. KJ2020A0358, and the young and middle-aged top talents training program of Anhui Polytechnic University.

## Declarations

**Conflict of interest** The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your

intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

- Wang CY, Bochkovskiy A, Liao HYM (2021) Rethinking BiSeNet for real-time semantic segmentation. *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit* 2:9711–9720. <https://doi.org/10.1109/CVPR46437.2021.00959>
- Huynh C, Tran AT, Luu K et al (2021) Progressive semantic segmentation. *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit* 1:16750–16759. <https://doi.org/10.1109/CVPR46437.2021.01648>
- Gwilliam M, Teuscher A, Anderson C et al (2021) Fair comparison: quantifying variance in results for fine-grained visual categorization. In: *Proc—2021 IEEE Winter Conf Appl Comput Vision, WACV 2021*, pp 3308–3317. <https://doi.org/10.1109/WACV48630.2021.00335>
- Mafra A, Dey S, Biten AF et al (2021) Multi-modal reasoning graph for scene-text based fine-grained image classification and retrieval. In: *Proc—2021 IEEE Winter Conf Appl Comput Vision, WACV 2021*, pp 4022–4032. <https://doi.org/10.1109/WACV48630.2021.00407>
- Gong X, Xia X, Zhu W et al (2021) Deformable gabor feature networks for biomedical image classification. In: *Proc—2021 IEEE winter conf appl comput vision, WACV 2021*, pp 4003–4011. <https://doi.org/10.1109/WACV48630.2021.00405>
- Chen Q, Wang Y, Yang T et al (2021) You only look one-level feature. *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit*. <https://doi.org/10.1109/CVPR46437.2021.01284>
- Sun P, Zhang R, Jiang Y et al (2021) Sparse R-CNN: end-to-end object detection with learnable proposals. *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit*. <https://doi.org/10.1109/CVPR46437.2021.01422>
- Wang CY, Bochkovskiy A, Liao HYM (2021) Scaled-yolov4: scaling cross stage partial network. *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit*. <https://doi.org/10.1109/CVPR46437.2021.01283>
- Qiao S, Zhu Y, Adam H et al (2021) VIP-DeepLab: learning visual perception with depth-aware video panoptic segmentation. *Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit*. <https://doi.org/10.1109/CVPR46437.2021.00399>
- Naseer M, Ranasinghe K, Khan S et al (2021) Intriguing properties of vision transformers. no. *NeurIPS*, [Online]. arXiv:2105.10497
- Radford A, Kim JW, Hallacy C et al (2021) Learning transferable visual models from natural language supervision. [Online]. arXiv:2103.00020
- Furfiari(tony) FA (2002) The transformer. *IEEE Ind Appl Mag* 8:8–15. <https://doi.org/10.1109/2943.974352>
- Cao H, Wang Y, Chen J et al (2021) Swin-Unet: Unet-like pure transformer for medical image segmentation, pp 1–14. [Online]. arXiv:2105.05537
- Dosovitskiy A, Beyer L, Kolesnikov A et al (2020) An image is worth 16 × 16 words: transformers for image recognition at scale. [Online]. arXiv:2010.11929
- Wang Y, Wu H, Zhang J et al (2021) PredRNN: a recurrent neural network for spatiotemporal predictive learning. 1–14.
- He K, Chen X, Xie S, et al (2022) Masked Autoencoders Are Scalable Vision Learners. 15979–15988. <https://doi.org/10.1109/cvpr52688.2022.01553>
- Chougule S, Koznek N, Adam G et al (2019) Reliable multilane detection and classification by utilizing CNN as a regression network. In: *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)*, pp 740–752
- Garnett N, Cohen R, Pe’Er T et al (2019) 3D-LaneNet: end-to-end 3D multiple lane detection. In: *Proceedings of the IEEE international conference on computer vision*, pp 2921–2930
- Su J, Chen C, Zhang K et al (2021) Structure guided lane detection. In: *IJCAI international joint conference on artificial intelligence*, pp 997–1003
- Gansbeke WV, Brabandere BD, Neven D et al (2019) End-to-end lane detection through differentiable least-squares fitting. In: *Proceedings—2019 international conference on computer vision workshop, ICCVW 2019*, pp 905–913
- Borkar A, Hayes M, Smith MT (2011) Polar randomized hough transform for lane detection using loose constraints of parallel lines. In: *ICASSP, IEEE international conference on acoustics, speech and signal processing—proceedings*, pp 1037–1040
- Yoo S, Seok L, Myeong H et al (2020) End-to-end lane marker detection via row-wise classification. In: *IEEE computer society conference on computer vision and pattern recognition workshops*, pp 4335–4343
- Aly M (2008) Real time detection of lane markers in urban streets. In: *IEEE intelligent vehicles symposium, proceedings*, p 7–12
- McCall JC, Trivedi MM (2006) Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation. *IEEE Trans Intell Transp Syst* 7:20–37
- Zhou S, Jiang Y, Xi J et al (2010) A novel lane detection based on geometrical model and Gabor filter. In: *IEEE intelligent vehicles symposium, proceedings*. IEEE, pp 59–64
- Selver MA, Er E, Belenlioglu B et al (2016) Camera based driver support system for rail extraction using 2-D Gabor wavelet decompositions and morphological analysis. *IEEE Int Conf Intell Rail Transp ICIRT 2016*:270–275. <https://doi.org/10.1109/ICIRT.2016.7588744>
- Zheng F, Luo S, Song K et al (2018) Improved lane line detection algorithm based on Hough transform. *Pattern Recognit Image Anal* 28:254–260. <https://doi.org/10.1134/S1054661818020049>
- Hur J, Kang SN, Seo SW (2013) Multi-lane detection in urban driving environments using conditional random fields. In: *IEEE intelligent vehicles symposium, proceedings*, pp 1297–1302
- Badrinarayanan V, Kendall A, Cipolla R (2017) SegNet: a deep convolutional encoder–decoder architecture for image segmentation. *IEEE Trans Pattern Anal Mach Intell*. <https://doi.org/10.1109/TPAMI.2016.2644615>
- Weng W, Zhu X (2021) INet: convolutional networks for biomedical image segmentation. *IEEE Access* 9:16591–16603. <https://doi.org/10.1109/ACCESS.2021.3053408>
- Wang Z, Ren W, Qiu Q (2018) LaneNet: real-time lane detection networks for autonomous driving. [Online]. arXiv:1807.01726
- Liu R, Yuan Z, Liu T et al (2021) End-to-end lane shape prediction with transformers. In: *Proceedings—2021 IEEE winter conference on applications of computer vision, WACV 2021*, pp 3693–3701
- Tabelini L, Berriel R, Paixão TM et al (2021) Keep your eyes on the lane: real-time attention-guided lane detection. In: *Proceedings of the IEEE computer society conference on computer vision and pattern recognition*, pp 294–302
- Hou Y, Ma Z, Liu C et al (2019) Learning lightweight lane detection CNNs by self attention distillation. In: *Proceedings of the IEEE international conference on computer vision*, pp 1013–1021
- Qin Z, Wang H, Li X (2020) Ultra fast structure-aware deep lane detection. In: *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)*, pp 276–291

36. Lo SY, Hang HM, Chan SW et al (2019) Multi-class lane semantic segmentation using efficient convolutional networks. In: IEEE 21st international workshop on multimedia signal processing, MMSP 2019
37. Li J, Mei X, Prokhorov D et al (2017) Deep neural network for structural prediction and lane detection in traffic scene. *IEEE Trans Neural Networks Learn Syst* 28:690–703. <https://doi.org/10.1109/TNNLS.2016.2522428>
38. Goodfellow I, Pouget-Abadie J, Mirza M et al (2020) Generative adversarial networks. *Commun ACM* 63:139–144. <https://doi.org/10.1145/3422622>
39. Ghafoorian M, Nugteren C, Baka N et al (2019) EL-GAN: embedding loss driven generative adversarial networks for lane detection. In: *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)*, pp 256–272
40. Liu Z, Lin Y, Cao Y et al (2021) Swin transformer: hierarchical vision transformer using shifted windows. In: *Proceedings of the IEEE international conference on computer vision*, pp 9992–10002
41. Neven D, De Brabandere B, Georgoulis S et al (2018) Towards end-to-end lane detection: an instance segmentation approach. In: *IEEE intelligent vehicles symposium, proceedings*, pp 286–291
42. Shi X, Chen Z, Wang H et al (2015) Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In: *Advances in neural information processing systems*, pp 802–810
43. Cho K, Van Merriënboer B, Gulcehre C et al (2014) Learning phrase representations using RNN encoder–decoder for statistical machine translation. In: *EMNLP 2014—2014 conference on empirical methods in natural language processing, proceedings of the conference*, pp 1724–1734
44. Ruder S (2016) An overview of gradient descent optimization, pp 1–14
45. Paszke A, Chaurasia A, Kim S et al (2016) ENet: a deep neural network architecture for real-time semantic segmentation, pp 1–10
46. He K, Zhang X, Ren S et al (2016) Deep residual learning for image recognition. In: *Proceedings of the IEEE computer society conference on computer vision and pattern recognition*, pp 770–778
47. Jia Deng, Wei Dong, Socher R et al (2009) ImageNet: a large-scale hierarchical image database. *IEEE*, pp 248–255
48. Kingma DP, Ba JL (2015) Adam: a method for stochastic optimization. In: *3rd International conference on learning representations, ICLR 2015—conference track proceedings*, pp 1–15

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.