Pattern Recognition 104 (2020) 107268

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Video semantic segmentation via feature propagation with holistic attention☆

Junrong Wu[a], Zongzheng Wen[a], Sanyuan Zhao[a,*], Kele Huang[b]

[a] Beijing Laboratory of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, 100081, PR China
[b] University of Chinese Academy of Sciences, Beijing, China

ARTICLE INFO

Article history:
Received 7 June 2019
Revised 9 January 2020
Accepted 10 February 2020
Available online 11 February 2020

Keywords:
Real-time
Attention mechanism
Feature propagation
Video semantic segmentation

ABSTRACT

Since the frames of a video are inherently contiguous, information redundancy is ubiquitous. Unlike previous works densely process each frame of a video, in this paper we present a novel method to focus on efficient feature propagation across frames to tackle the challenging video semantic segmentation task. Firstly, we propose a Light, Efficient and Real-time network (denoted as LERNet) as a strong backbone network for per-frame processing. Then we mine rich features within a key frame and propagate the across-frame consistency information by calculating a temporal holistic attention with the following non-key frame. Each element of the attention matrix represents the global correlation between pixels of a non-key frame and the previous key frame. Concretely, we propose a brand-new attention module to capture the spatial consistency on low-level features along temporal dimension. Then we employ the attention weights as a spatial transition guidance for directly generating high-level features of the current non-key frame from the weighted corresponding key frame. Finally, we efficiently fuse the hierarchical features of the non-key frame and obtain the final segmentation result. Extensive experiments on two popular datasets, i.e. the CityScapes and the CamVid, demonstrate that the proposed approach achieves a remarkable balance between inference speed and accuracy.

1. Introduction

Semantic segmentation is a fundamental task in the area of computer vision, with the goal of assigning a label to each pixel in a given scene. Inspired by deep learning, this task has achieved great progress. Most of the existing works tend to parse static scenes, however, the challenging video-based task remains to be explored, which has a wide range of applications, i.e. autonomous driving, robotics, surveillance, and etc.

One of the bottlenecks of video semantic segmentation task is information redundancy. Videos have naturally rich spatio-temporal information, which requires much more computing resources than images. However, consecutive video frames typically vary very little while having high similarity and coincidence, which is easily neglected by existing works. Directly applying an image-based approach to each frame of a video neglects critical time-domain context information, and tends to cause inconsistently predictions and ignore those small but active-across-frame objects which can be inferred with temporal information. Inference speed is another bottleneck of video semantic segmentation task. As far as we know, most previous methods process video frames slowly and there is no existing network to achieve real-time speed yet. However, real-time inference speed is necessary for practical applications like autonomous driving.

To cope with bottleneck problems of the video semantic segmentation task, previous works mainly followed two thoughts. One is to aggregate information among multiple frames [1], and the other one is to propagate features across frames. Specifically, the former one usually relied on methods with temporal or sequential nature, such as three-dimensional(3D) convolution [2,3] and recurrent neural network (RNN) [4,5]. These methods took the entire frame-wise information as input, thus suffering from high computational cost. Instead, the latter one attempted to fully leverage across-frame consistency features by propagating them along the temporal dimension. Our methodology follows this direction to efficiently reduce information redundancy. In order to propagate features, previous works [5,6] usually employ a separate optical flow network, which costs extra time and often causes motion blur. The work [7] proposed transition layers in deconvolutional network to maintain spatial and temporal consistency of predicted results. Yet, such method is not suitable for processing large-scale images.

☆ This work was supported by the National Natural Science Foundation of China under Grant 61902027.

* Corresponding author.
E-mail address: zhaosanyuan@bit.edu.cn (S. Zhao).

https://doi.org/10.1016/j.patcog.2020.107268
0031-3203/

To solve the existing problems, in this paper, we firstly propose a Light, Efficient and Real-time network (denoted as LERNet) as a strong backbone network of our video semantic segmentation method. Our LERNet applies encoder-decoder architecture, and we propose the Residual Double-branched Depthwise Separable convolution block (RDDS block) in the encoder to efficiently capture detail information and effectively reduce computation. To realize feature propagation, we utilize the key frame scheduling and propose a unique Temporal Holistic Attention module (THA module) to indicate spatial correlations between a non-key frame and its previous key frame. More specifically, we construct a real-time fully convolutional network with our proposed attention-based feature propagation architecture. Firstly, input frames are divided into key frames and non-key frames according to fixed key frame selection scheduling. As for key frames, we employ the whole backbone structure to hierarchically derive rich spatial information for feature propagation. Instead of wasting much time to extract redundant features through the whole backbone network, the following non-key frames merely need to extract their low-level features and maintain spatial details by the shallow layers of our backbone network, then we fuse low-level features with the attention-weighted high-level features propagated by the previous key frame. To efficiently propagate high-level features, we present an attention-based method from a unique view. Concretely, we take low-level feature maps of both the non-key frame and its corresponding key frame as input, and obtain the holistic attention map by calculating their spatial similarities between any two positions of the feature maps, and the value of each position represents the degree of affinity. Since the attention map integrates per-pixel correlations between two frames, it can be viewed as a spatial transition guide to capture inter-frame consistency information. The high-level features of the non-key frame can be obtained by applying the attention weights to high-level features of the corresponding key frame. Then we fuse it with the low-level features of the non-key frame to complement new information that does not exist in the previous key frame, thus enhancing the capability of handling complicated and ever-changing scenarios. The proposed model is conveniently differentiable and end-to-end trainable.

Our main contributions can be summarized into four aspects. (1) We propose LERNet for real-time image semantic segmentation and use it as our backbone network. Our LERNet can process high-resolution images (512Ã1024) at 100 fps on a single GTX 1080Ti card and get 69.5% Mean IoU on the Cityscapes dataset with merely 0.65 M parameters. (2) We focus on efficient feature propagation across redundant video frames and propose a novel methodology for video semantic segmentation to capture temporal consistency features. (3) We devise the Temporal Holistic Attention module to capture spatial correlations between key frames and non-key frames, and efficiently propagate features. (4) Our extensive experiments on street scene datasets, including the CityScapes [8] and the CamVid [9], demonstrate that our proposed model achieves real-time inference speed while ensuring high accuracy. In particular, our model can perform at the speed of 131 fps on the CityScapes dataset.

2. Related work

2.1. Real-time image semantic segmentation

Real-time image semantic segmentation methods aim at greatly improving inference speed while ensuring accuracy. For this sake, it is important to reduce model weight and extract rich features. ENet [10] was the first network designed for real-time semantic segmentation task, it employed an extremely shallow and light network to save computation and achieve speedup. ESPNet [11] utilized new spatial pyramid modules to achieve efficient computation. ERFNet [12] used new layers with residual connections and factorized convolutions to get a trade-off of speed and accuracy. ICNet [13] designed a cascade structure to improve efficiency and takes multi-scale images as input. BiSeNet [14] used two paths to obtain spatial detail information and semantic context information, respectively. LEDNet [15] designed a lightweight encoder-decoder network which utilized channel split and shuffle in each residual block to reduce computation cost. Different from these methods, we propose to use two branches in our convolutional block and enlarge the number of channels of each branch to fully exploit detail information. And we employ depthwise separable convolutions in each branch to save computational cost.

2.2. Video semantic segmentation

Many image semantic segmentation networks have extended the great success of the FCN [16]. However, the challenging video semantic segmentation task still remains to be explored. Since there are strong correlations between adjacent video frames [17,18], information redundancy is a common problem. Directly applying the image-based method to each frame of the video is time-consuming and not able to fully utilize the cross-frame relationships, resulting in unsatisfied performance. Previous efforts can be approximately classified according to how to utilize temporal information across frames. Some works like [2,3] encoded the motion and structure features by employing 3D convolution, which can be viewed as an information aggregation way to take the entire frame-wise information as input, thus not efficient enough. Similarly, works like [4,5,19] employed recurrent neural networks to aggregate frame-wise information and existed similar drawbacks as 3D convolution based methods. In addition, several methods as [20,21] used CRF to model spatial and temporal context, which suffered from high computational cost because of the expensive inference of CRF. Besides, Nilsson and Sminchisescu [5], Zhu et al. [6], Wang et al. [22] employed a separate network to calculate optical flow for propagating features from the previous frame to the current one. However, accurate optical flow estimation is difficult to achieve, time-consuming and always misaligned. Instead of optical flow, Shelhamer et al. [23] adapted multi-stages FCN and directly reused the second or third stage features of previous frames to save computation, but straightforward replication is not robust enough to tackle significant changes in the scene. Li et al. [24] learned a threshold to select key frames and propagates features from key frames to others through spatially variant convolution kernels. Wang et al. [7] proposed transition layers in deconvolutional network to maintain spatial and temporal consistency of predicted results.

Some other works [25–27] have different goals, for instance, Jin et al. [27] used GAN to first predict future video frames in an unsupervised manner and then use the synthetic “labels” for video semantic segmentation.

2.3. Attention mechanism

The attention mechanism is one of the most important characteristics of the human visual system, which is capable to capture the most important features in a scene. Machine attention was first applied to areas such as machine translation, and image captioning, etc. When applied to the field of computer vision, attention can effectively emphasize the image positions that human eyes focus on. Some tasks like fixation prediction [28,29] and video object segmentation [30] naturally utilize attention. In the area of video semantic segmentation task, the use of attention mechanism is rare. Some image-based methods exploit spatial or/and channel attention to obtain rich context and improve results. SENet [31] proposed to squeeze features before recovering them. Wang

Table 1
Detailed structure of the proposed LERNet.

Part	Operator	Output size
Encoder	Downsampling	32 × 256 × 512
	3 × RDDS-block	32 × 256 × 512
	Downsampling	64 × 128 × 256
	2 × RDDS-block	64 × 128 × 256
	Downsampling	128 × 64 × 128
	RDDS-block	128 × 64 × 128
	RDDS-block (dilation = 2)	128 × 64 × 128
	RDDS-block (dilation = 5)	128 × 64 × 128
	RDDS-block (dilation = 9)	128 × 64 × 128
	RDDS-block (dilation = 2)	128 × 64 × 128
	RDDS-block (dilation = 5)	128 × 64 × 128
	RDDS-block (dilation = 9)	128 × 64 × 128
	RDDS-block (dilation = 17)	128 × 64 × 128
Decoder	1 × 1 Conv	C × 64 × 128
	Upsampling (× 8)	C × 512 × 1024

et al. [32] utilized the non-local idea to build blocks for capturing long-range dependencies. CCNet [33] and DANet [34] obey the similar thought of considering the pair-wise similarities among all the pixels in a self-attention manner. Inspired by above mentioned self-attention methods, our work use the attention mechanism to capture correlations between two video frames, and treat the attention matrix as an efficient transition guidance from a unique perspective.

3. Proposed methodology

3.1. Overview

In this paper, we mainly focus on two aspects to tackle the challenging video semantic segmentation task. The first one is how to accelerate the processing of a single video frame while maintaining the accuracy of segmentation. And the second one is how to reduce inter-frame information redundancy between key and non-key frames. To guarantee the processing speed and prediction accuracy of video semantic segmentation task, we design a novel Light, Efficient and Real-time network (denoted as LERNet) as a strong baseline. To reduce information redundancy between consecutive video frames, the direct propagation of the consistency information between the two frames is required, so we devise the temporal holistic attention module to achieve real-time feature propagation.

Given a general description, we firstly introduce the novel backbone network in Section 3.2, and then explain the feature propagation component of our model in detail, namely the temporal holistic attention module in Section 3.3. Finally, we will describe the feature propagation process in Section 3.4.

3.2. Proposed backbone network: LERNet

To process single frames more efficiently, we propose a strong image-based backbone network named LERNet before tackling the video semantic segmentation task. As is illustrated in Table 1, our network employs the encoder-decoder architecture, where the encoder extracts image features, aiming at maintaining rich spatial detail information and obtaining hierarchical global semantic information. The decoder mainly reduces channel numbers and upsamples the downsampled result of the encoder to the original resolution.

As for encoder, we propose a novel block named Residual Double-branched Depthwise Separable convolution block (RDDS block), whose structure is illustrated as Fig. 1. The encoder is mainly stacked by RDDS blocks. Residual module which constitutes existing networks (like ResNet) has already been proven effective in various areas, however, it has two main disadvantages. Firstly, ResNet tends to make a deep model with more than 100 layers because it can increase the receptive field and capture more complex features. However, higher number of layers also bring increasing runtime and memory requirements [35]. Secondly, stacking traditional convolution layers is inefficient, and the network is computationally intensive. In response to the above problems, some methods have optimized their network architecture. For example, LEDNet [15] employs 1 × 3 and 3 × 1 convolutions instead of 3 × 3 convolution to reduce the parameter amount, and equally splits the channel number into two branches for following shuffle operation. However, the number of channels in each convolution is small (maximum is 64), and the parameter amount can be further reduced.

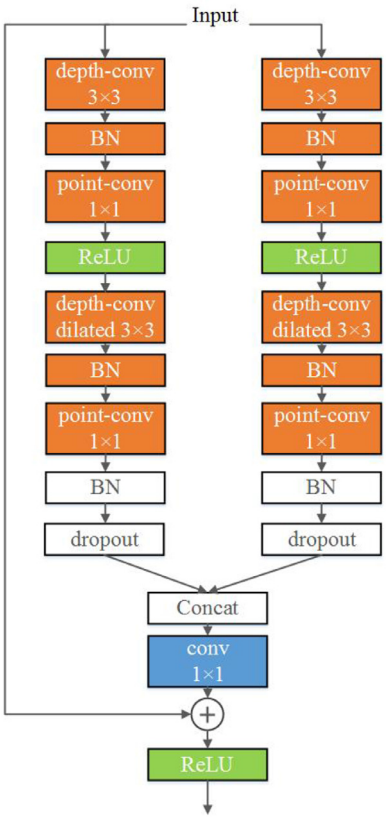

Fig. 1. The Residual Double-branched Depthwise Separable convolution block (RDDS block).

We designed the RDDS block to capture rich features and further reduce computational cost as well as parameter amount. Utilizing two depthwise separable convolutions in each branch saves the model's parameter amount and computational cost, which in turn keeps more feature channels (maximum is 128) in each branch for calculation under the same hardware constraints.

As shown in Fig. 1, two branches are symmetrically distributed in the RDDS block. In each branch, a fully sampled 3 × 3 depthwise convolutional layer is used to compute input features intensively and obtain local area information, then we use a batch normalization layer and a 1 × 1 point-wise convolutional layer to fuse the features of each channel. Next, we utilize a ReLU layer to increase network capacity, and adopt a 3 × 3 dilated depthwise convolutional layer to increase the receptive field and get global context information. After the dilated depthwise convolutional layer, we also employ a batch normalization layer and a point-wise convolutional layer. We add a dropout layer at the end of each branch to prevent overfitting. To be added with the input, the results of

Table 2
Comparison between different decoders on the CityScapes validation set. Both the GFLOPs and Frames(fps) are estimated under the spatial size of 512 × 1024.

Method	GFLOPs	Parameter(M)	Frames(fps)	mIoU(%)
LERNet(Ours)	12.76	0.65	100	67.67
encoder+LEDNet [15] decoder	12.78	0.662	90	67.24
encoder+Deeplab v3+ [39] decoder	14.70	0.67	83	67.7

the two branches are concatenated, and we reduce the number of channels of the concatenated result by a 1 × 1 convolution. Finally, we utilize a ReLU layer. In addition to RDDS blocks, the encoder also has downsampling modules, which are implemented by the concatenation of a max-pooling with stride 2 and a 3 × 3 convolution.

To compare the $K \times 1$ and $1 \times K$ convolutions used in [15] with the depthwise seperable convolution with $K \times K$ kernel used by us, we assume there are $C_{in} \times H \times W$ input and $C_{out} \times H \times W$ output, parameter amount and computational cost of the former are $2C_{in}KC_{out}$ and $2C_{in}KC_{out}HW$ respectively, while parameter amount and computational cost of the latter are $C_{in}K^2 + C_{in}C_{out}$ and $C_{in}K^2HW + C_{in}C_{out}HW$ respectively. For practical implementation, if $C_{in} = C_{out} = 64, K = 3, H = 256, W = 128$, then both parameter amount and computational amount of our depthwise seperable convolution are less than 1/5 of those of convolutions used in [15]. Our RDDS block uses a double-branched structure to obtain richer detail features, and we designed comparison experiment with single-branched structure to prove the effectiveness of our double-branched structure, details will be shown in Section 4. Operating on more channels can retain more spatial details in the low-level layers and obtain richer semantic features in the high-level layers, while compared with [15], the computational amount of our encoder network is slightly increased. To further compensate for the spatial detail information, the input to the entire module is summed with the concatenated result of the two branches by a shortcut connection.

The main purpose of the decoder is to upsample the reduced feature map of the encoder to the input size. We have designed different decoder schemes for experimental comparison, and the experimental results are shown in Table 2, which will be introduced in Section 4. We find that complicated decoders have little effect on the improvement of accuracy but seriously slow down the inference speed. Hence, we employ a simple decoder, which only consists of a 1 × 1 convolution and a upsampling module.

3.3. Temporal holistic attention module

Since we have a strong backbone network for per-frame processing, now we cope with the video semantic segmentation task. To fully utilize across-frame consistency, to reduce information redundancy between adjacent video frames and to further save inference time, we focus on the feature propagation from a novel perspective. Specifically, we propose the Temporal Holistic Attention module (THA module) to perform feature propagation between key frames and the following non-key frames. After deeply mining the spatial correlations between the low-level feature maps of the key frame and the non-key frame, we utilize the THA module to generate an attention map, which implicitly contains across-frame consistency information and can be viewed as a transition guidance for feature propagation. In this section, we will introduce how to calculate the attention map in detail, Fig. 2 shows the whole process.

Extracted after the second downsampling block of our proposed LERNet encoder, we denote $\{\mathbf{F}_l^n, \mathbf{F}_l^k\} \in \mathbb{R}^{C \times H \times W}$ (C × H × W=128 × 64 × 128) as low-level features of the current non-key frame and its related key frame, respectively. Where C represents the number of feature channels, and H and W denote the spatial size. In order to obtain the correlation between each pixel in two adjacent feature maps, we view $\mathbf{F}_l^k$ as a set $\mathbb{V}$. We denote $\mathbb{V}$ as $\{\mathbf{V}_i, i = 1, 2, \ldots, N\}$, which is composed of $N = H \times W$ vectors with the scale of $C \times 1 \times 1$, and each vector $\mathbf{V}_i$ contains all the channel information of the corresponding position *i*. Then $\mathbf{V}_i$ and $\mathbf{F}_l^n$ are multiplied to obtain a correlation map with the scale of $1 \times H \times W$. Hence, by operating on each position of the feature map $\mathbf{F}_l^k$, we can get a set of N correlation maps with the scale of $1 \times H \times W$ and represent it as tensor $\mathbf{A} \in \mathbb{R}^{N \times H \times W}$, which contains spatial similarity information between all positions of $\mathbf{F}_l^n$ and $\mathbf{F}_l^k$. The more similar the ith position of $\mathbf{F}_l^n$ is to the jth position of $\mathbf{F}_l^k$, the larger the corresponding value of the ith spatial position on the jth channel of the tensor $\mathbf{A}$. So we feed $\mathbf{A}$ to two parallel branches to get maximum point-to-point responses, where $\mathbf{A}$ is average-pooled and maximum-pooled in the channel dimension. Then the results of the two branches are concatenated to obtain the maximum response attention map with channel number 2. We use a 5 × 5 convolutional layer to reduce the number of channels to 1. Then we use a Sigmoid activation layer to limit the value to $[-1, 1]$, and finally get the attention map $\mathbf{W}_a$ with the scale of $1 \times H \times W$.

In practical implementation, the specific dimension-reduced multiplication process between $\mathbf{F}_l^n$ and $\mathbf{F}_l^k$ is as shown in Fig. 2. To save computational cost, a 1 × 1 convolution layer is first applied to obtain the features after dimension reduction, namely $\{\mathbf{F}^{nl}, \mathbf{F}^{kl}\} \in \mathbb{R}^{C' \times H \times W}$, where $C' = 1/32C = 4$. Then we reshape both $\mathbf{F}^{nl}$ and $\mathbf{F}^{kl}$ to $\mathbb{R}^{C' \times N}$, where N equals to $H \times W$. And after a transpose operation, the scale of key frame feature map becomes $N \times C'$.

Specifically, at each row i of feature map $\mathbf{F}^{kl}$, we can get a position vector $\mathbf{F}_i^{kl} \in \mathbb{R}^{C'}$, which contains long-range information of all channels at each position on the spatial dimension. Similarly, each column j of feature map $\mathbf{F}^{nl}$ can be represented as a vector $\mathbf{F}_j^{nl} \in \mathbb{R}^{C'}$. By a matrix multiplication operation, we can get each attention element $\mathbf{A}_{i,j}$, which represents the contextual correlation between each position pair:

$$\mathbf{A}_{i,j} = \mathbf{F}_i^{kl}(\mathbf{F}_j^{nl})^\top, \tag{1}$$

where $i, j \in [1, N]$, and $\mathbf{A} \in \mathbb{R}^{N \times N}$. Then we reshape $\mathbf{A}$ into $N \times H \times W$ and perform subsequent operation. Max-pooling and average-pooling are performed in parallel to compress information along the channel dimension. We get the attention weight map $\mathbf{W}_a$ after a 5 × 5 convolutional layer to reduce channel dimension and a Sigmoid layer to limit the value of each element:

$$\mathbf{W}_a = Sigmoid(Conv(Concat(Maxpool(\mathbf{A}), Avgpool(\mathbf{A})))), \tag{2}$$

where $\mathbf{W}_a \in \mathbb{R}^{1 \times H \times W}$, and each element of $\mathbf{W}_a$ is a scalar.

Unlike [32], we take feature maps derived from two different frames as inputs, and each element in the resulting attention map is a scalar, greatly reducing the computational cost of subsequent calculations. Since spatial sizes (64 × 128) and channel dimensions (128/32=4) of two inputs are small, main computational cost of the THA module (namely the matrix multiplication) is not very expensive (about 0.27G). Since we already get the transition guidance, next we use it to efficiently realize feature propagation.

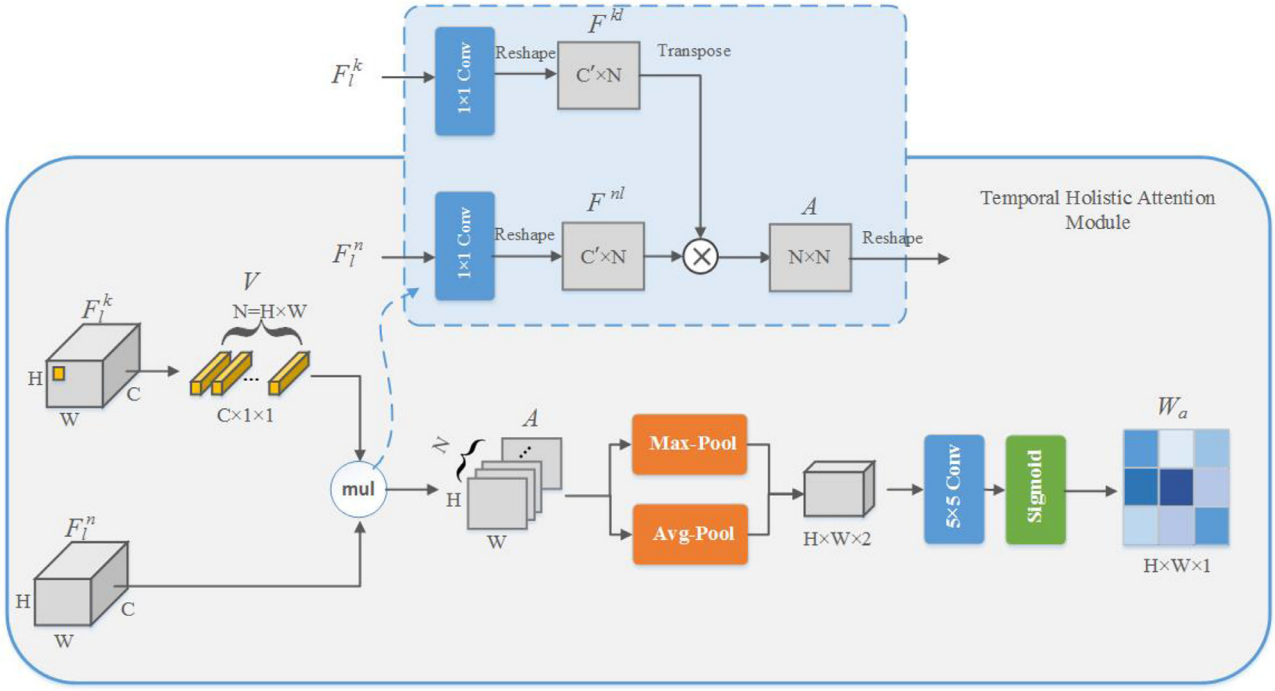

Fig. 2. The details of the Temporal Holistic Attention module.

3.4. Feature propagation with attention guidance

Our goal is to find an efficient way to propagate high-level features from accurately calculated key frames to their following non-key frames, and we have already acquired an attention guidance $\mathbf{W}_a$. Then we perform element-wise multiplication between $\mathbf{W}_a \in \mathbb{R}^{1\times H\times W}$ and our proposed LERNet encoder's final output feature map of the key frame, namely the high-level feature map $\mathbf{F}_h^k \in \mathbb{R}^{C\times H\times W}$, to get the synthetic high-level feature map of the current non-key frame $\mathbf{F}_h^n \in \mathbb{R}^{C\times H\times W}$:

$$\mathbf{F}_h^n = \mathbf{W}_a \odot \mathbf{F}_h^k, \tag{3}$$

where $\odot$ represents element-wise point multiplication. Note that the low-level and high-level features of our backbone network have the same channel dimension and spatial size. After element-wise multiplications, each position of $\mathbf{F}_h^n$ implicitly contains contextual information from all positions of the key frame, and preserves similar features, namely the across-frame consistency features, between two frames at any scales from a global view. Since each element of $\mathbf{W}_a$ is a scalar, our proposed method requires much lower computation cost compared with other feature propagation method like [32]. For instance, assuming that the input scale is $C \times H \times W$, our method only requires HWC times multiplication to get the result. In contrast, [32] performs matrix multiplication between reshaped input $(C \times N, N = H \times W)$ and the attention map $(N \times N)$, which requires $(HW)^2C$ times multiplication. For $H = 128, W = 64$, our method reduces more than 8000 times the amount of calculation.

After multiplication with the attention map, the synthetic high-level features are fused with original low-level features to complement new information of the non-key frame. Since channel dimensions of the high-level and low-level feature maps are the same, we directly add them and get the final high-level feature map of the current non-key frame, denoted as $\mathbf{F}^n$:

$$\mathbf{F}^n = \mathbf{F}_l^n + \mathbf{F}_h^n. \tag{4}$$

After obtaining $\mathbf{F}^n$, the feature propagation task is successfully achieved. For tackling the video semantic segmentation task, Fig. 3 illustrates the overview pipeline of our framework.

As shown in Fig. 3, unlike previous works that make great efforts to perform frame-by-frame inference, our model only applies the entire deep backbone network to key frames, and efficiently propagates high-level features of key frames to their following non-key frames. We employ our proposed fully convolutional encoder-decoder network LERNet as the backbone to extract features from video frames. To distinguish low-level and high-level features, we divide the backbone into two parts, from the beginning to the second downsampling in the LERNet encoder is regarded as the low-level network P_l and the latter part of the encoder is viewed as the high-level network P_h. Selected by the key frame scheduling, key frames obtain hierarchical features $\mathbf{F}_l^k, \mathbf{F}_h^k$ from both P_l and P_h, while non-key frames only need P_l to obtain low-level features $\mathbf{F}_l^n$. Then we feed $\mathbf{F}_l^k$ and $\mathbf{F}_l^n$ as inputs to the proposed temporal holistic attention module, which will generate an affinity attention matrix to indicate per-pixel spatial correlations between the non-key frame and its previous key frame. Next, we perform element-wise multiplication between the attention matrix and $\mathbf{F}_h^k$ to directly acquire the predicted high-level feature map of the non-key frame $\mathbf{F}_h^n$. Namely, we use the attention matrix as a spatial guidance to adaptively transfer temporal consistency information from key frame to non-key frame. After that, we add the original $\mathbf{F}_l^n$ and the predicted $\mathbf{F}_h^n$. Finally, we obtain per-frame segmentation result after decoder.

4. Experiments

In this part, we extensively experiment on two challenging datasets, i.e. the CityScapes and the CamVid. In this section, we first introduce the datasets and evaluation metrics, then elaborates on the implementation details over all the experiments. We design diverse experiments to confirm the effectiveness and efficiency of our proposed LERNet. Finally, to perform experiments on the video semantic segmentation task, we estimate the effect of our pro-

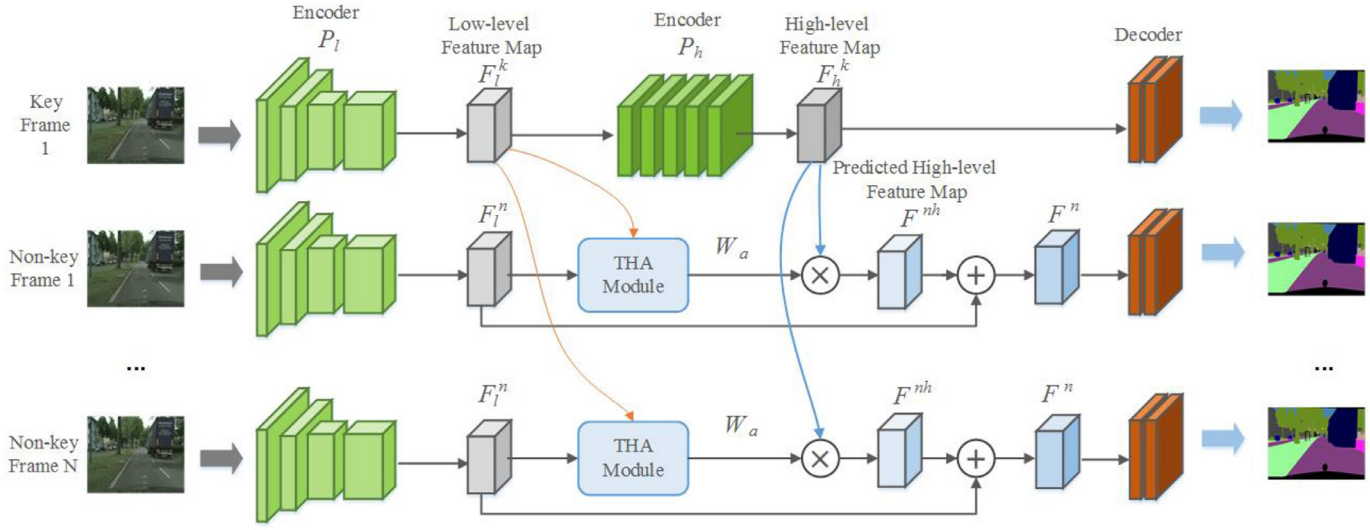

Fig. 3. The overall pipeline of our proposed architecture. For a key frame decided by the selection scheduling, it will pass through the complete backbone network including P_l and P_h to get hierarchical features $\mathbf{F}_l^k$, $\mathbf{F}_h^k$. And the following non-key frame only needs the low-level network P_l to obtain low-level features $\mathbf{F}_l^n$. Our proposed temporal holistic attention module take $\mathbf{F}_l^k$ and $\mathbf{F}_l^n$ as input to generate the weight matrix $\mathbf{W}_a$, which will be applied to $\mathbf{F}_h^k$ and obtain predicted high-level features $\mathbf{F}^{nh}$ for the current non-key frame. Then the feature fusion module adaptively fuses predicted features with the original low-level features of the current non-key frame and get $\mathbf{F}^n$. Finally, we apply different convolutional layers to two kinds of frames and get segmentation outputs. Note that colored curves represent feature propagation.

posed THA module and compare our method with existing state-of-the-art methods to verify our benefits.

4.1. Datasets and evaluation metrics

CityScapes. The CityScapes is a large urban street scene dataset containing video sequences collected from 50 different cities. For the scene parsing task, there are 5000 images with high-quality dense annotations, which are divided into sets of 2975, 500 and 1525 for training, validating and testing respectively. Besides, 20,000 roughly labeled images are provided as extra data. Totally 19 classes are used for training and testing. The resolution of each image is 2048 $\times$ 1024. For the continuous video frame dataset CityScape-sequence, each video clip contains 30 frames, of which only the 20th frame is labeled.

CamVid. The CamVid is another street view dataset that contains a total of 701 consecutive frames with annotations, of which 367 are for training, 101 for validation, and 233 for testing. The resolution is 960 $\times$ 720, and there are 11 common classes in total.

Evaluation Metrics. In this paper, two widely used evaluation metrics are employed to evaluate the performance of our model, namely the mean Intersection-over-Union (class IoU or mIoU) and Pixel Accuracy (PA). mIoU is the mean ratio of the intersection and union between the predicted and ground-truth values on all classes. PA is equal to the percentage of all correctly classified pixels.

4.2. Implementation details

Training Settings. During training, our model follows a two-stage training strategy. Due to the fact that manual annotations are expensive, video frames are usually sparsely labeled. Taking Cityscapes as an example, in each video clip, only the 20th frame out of 30 frames has the ground-truth label. To tackle the problem of lacking training supervision, we use the state-of-the-art image semantic segmentation method [36] to generate pseudo labels of CityScapes-sequence training set as a training supplement, which may enhance the robustness of our model. For CityScapes dataset, the backbone network LERNet is pre-trained on the CityScapes training set with fine labels and the CityScapes-sequence training set with pseudo labels in the per-frame manner. And then we train the entire network by using the much larger CityScapes-sequence dataset, which contains consecutive video frames to learn for long-term inter-frame consistency. When training the entire network, our main goal is to learn the feature propagation module (THA module). We utilize fixed key frame selection scheduling (1 key frame out of K=5 frames) and employ the similar training method of Li et al. [37], which takes a key frame and its following non-key frame as a pair. The steps between the key frame and its following non-key frame ranges from 1 to D (we set D=4). We save the low-level network output of key frame and propagate it via THA module to generate final output of the non-key frame. For CamVid dataset, the resolution of each image is 360 $\times$ 480 during training, validating and testing. Since we find that images of its three separate sets (i.e. train,val,test) are discontinuous, we combine training set with validation set for training the whole framework, and we reuse the backbone model pre-trained on CityScapes.

We use the stochastic gradient descent (SGD) optimization algorithm, momentum is 0.9 and weight decay is $1e^{-4}$. When training the backbone network LERNet, the initial learning rate is set to $5e^{-4}$, and when training the entire network, the initial learning rate of THA module is set to $3e^{-3}$ while that of backbone network is $2e^{-6}$. Follow [34,38], we use the "*poly*" learning rate strategy to multiply $(1-(iter/max\_iter)^{power})$ by the initial learning rate after each iteration, where *power* is 0.9. The proposed video semantic segmentation model is implemented with Pytorch, training on 4 GTX 1080Ti GPUs (11G Memory). And we set the batch size to 12.

Loss Function. We employ cross entropy loss function to help minimizing the difference between segmentation results of video frames and their corresponding ground-truth labels.

Data Augmentation. We employ several data augmentation methods during training, including random horizontal flip, random translation and random color jitter. As for random translation, the images and corresponding labels can simultaneously translate several pixels along the horizontal and vertical directions. Translation distance contains 0,1,2. As for random color jitter, we change

Table 3
Comparison between networks with our RDDS block and the single-branched block on CityScapes validation set for single frame prediction. Note that the single-branched block has exactly the same structure as each branch of our RDDS block.

Method	mIoU(%)	GFLOPs	Parameters(M)	Frames(fps)
Single-branched network	67.9	5.0	0.37	125
Double-branched network(Ours)	69.5	12.7	0.65	100

the brightness, contrast and saturation of an image, and the jitter scales of them are floats ranging from 0 to 2.

4.3. Experiments of LERNet

To show the benefits of our proposed backbone network, we did sufficient experiments. Firstly, we carry out a comparison experiment of different decoders. Second, to prove the effectiveness of our double-branched block (RDDS block), we compare it with single-branched block by visualized results and a table. Next, we perform a comparison experiment on different numbers of channels to verify the advantage of more channels. Besides, we compare our LERNet with other state-of-the-art real-time semantic segmentation methods to confirm the advantages of our method, and we draw a figure to intuitively display the performance of inference speed, FLOPs and mIoU.

4.3.1. Comparison of different decoders

As shown in Table 2, we compare our simple baseline decoder with other different decoders. Note that for these three methods, all experimental settings are the same except for the structure of decoder. The "LEDNet decoder" is just the same as that of Wang et al. [15]. The "Deeplab v3+ decoder" represents a decoder we designed, whose motivation is the same as that of Deeplab v3+, namely, fuse low-level and high-level features before upsampling to get better results. We save the output after the first downsampling block of our encoder as the low-level feature map, and view the final output of our encoder as the high-level feature map. After upsampling its feature map to the spatial size of the low-level feature map, the high-level feature map is concatenated with the low-level one. Finally, we use a 3×3 convolutional layer to reduce the channel dimension to the number of classes and use a upsampling block to match the input size. The performance of mIoU is estimated after 300 epochs training only on the CityScapes training set with fine labels. Both GFLOPs and inference speed are estimated under the spatial size of 512×1024. In all experiments about inference speed, we use a single GPU to measure and set the batch size of input images to 3. In Table 2, it can be found that our LERNet requires the least GFLOPs and parameters, and has the fastest speed. Although use the "Deeplab v3+ decoder" improves a little the performance of mIoU, the inference speed is much slower. Hence, as a trade-off result, we employ our simple decoder.

4.3.2. Evaluation of the double-branched structure

We perform a comparison experiment between our proposed RDDS block and the single-branched structure block on CityScapes validation set. The two methods are under the same conditions except for the number of branches. The single-branched block has exactly the same structure as each branch of our RDDS block. Table 3 shows that our double-branched method outperforms the single-branched one by 1.5% mIoU. The GFLOPs and parameter amount of the single-branched method are basically half, while the inference speed of the single-branched method is 25 fps faster than that of our double-branched method. The final prediction results of two methods are displayed in Fig. 4, as the red boxes emphasize, our method with double-branched structure is on the top left and predicts details better, which may because our method can capture richer detail features.

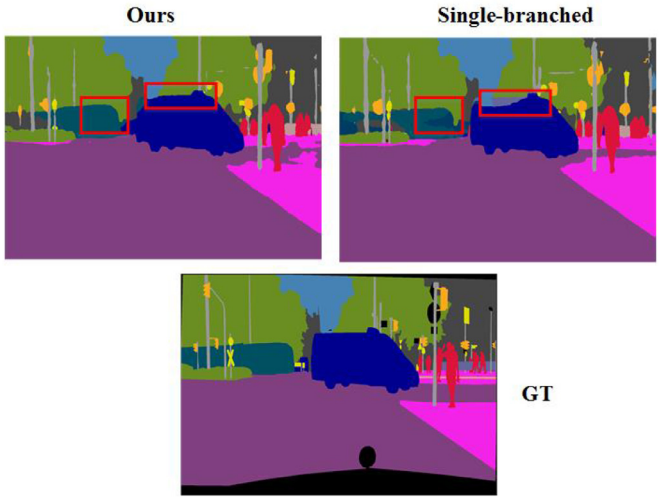

Fig. 4. Prediction results of our double-branched method (LERNet) and the single-branched method.

Table 4
Comparison on the number of channels of our LERNet on CityScapes validation set. Maximum Channels means the maximum number of channels in the encoder network. Note that two models are exactly the same except for the number of channels.

Maximum Channels	mIoU(%)	GFLOPs	Parameters(M)	Frames(fps)
64	53.1	3.6	0.17	100
128(Ours)	69.5	12.7	0.65	100

4.3.3. Comparison on the number of channels

To compare the effect of the number of channels, we halve the number of channels of each RDDS block and downsampling operator in the encoder. The maximum number of channels of each branch in RDDS block is 64. We train the new encoder-decoder network by using the same settings as our backbone network LERNet (except for the initial learning rate. Since the mIoU of the model with less channels increases much slower, we set the initial learning rate to $5e^{-3}$). As shown in Table 4, our network with 128 channels outperforms the network with 64 channels by 16.4% mIoU, which proves that more channels can get much better performance. We speculate that the reason may be that more channels contain more useful information without loss. The GFLOPs and parameter amount of the network with 64 channels are almost a quarter of those of our network with 128 channels, nevertheless, the inference speeds of these two networks are the same, which implies that the inference speed of a network has little to do with the number of channels.

4.3.4. Comparison with the state-of-the-art methods

We compare our method and several state-of-the-art semantic segmentation works to verify benefits of our method. As Table 5

Table 5
Comparison with existing state-of-the-art methods on CityScapes test and validation set for single frame prediction. "Time" represents inference time estimated on a single GPU.

Method	Input size	GFLOPs	Parameters(M)	Time(ms)	Frames(fps)	test mIoU(%)	val mIoU(%)	GPU
PSPNet [40]	713 × 713	412.2	250.8	1288	0.78	81.2	–	TitanX
DeepLab [41]	512 × 1024	457.8	262.1	4000	0.25	63.1	–	TitanX
SegNet [42]	640 × 360	286	29.5	60	16.7	57	–	TitanX
ENet [10]	640 × 360	3.8	0.4	7	135.4	58.3	–	TitanX
ESPNet [11]	512 × 1024	–	0.36	8.9	112	60.3	–	TitanX Pascal
LERNet(Ours)	768 × 1536	28.7	0.65	11	90.9	–	65.1	1080 Ti
ERFNet [12]	512 × 1024	–	2.1	24	41.7	68	70.0	TitanX
BiSeNet [14]	768 × 1536	14.8	5.8	9.4	105.8	68.4	69.0	TitanXp
ICNet [13]	1024 × 2048	28.3	26.5	33	30.3	69.5	67.7	TitanX
LERNet(Ours)	512 × 1024	12.7	0.65	10	100	66.5	69.5	1080 Ti

Table 6
Per-class IoUs and Mean IoUs on the CityScapes validation set for single frame prediction comparison. List of classes: Road, Sidewalk, Building, Wall, Fence, Pole, Traffic Light, Traffic Sign, Vegetation, Terrain, Sky, Pedestrian, Rider, Car, Truck, Bus, Train, Motorbike and Bicycle.

Methods	Roa	Sid	Bui	Wal	Fen	Pol	TLi	TSi	Veg	Ter
ERFNet [12]	97.4	80.6	90.3	55.8	50.1	57.5	58.6	68.2	90.9	61.2
LERNet(Ours)	97.4	79.9	89.3	55.6	52.0	47.3	55.0	64.0	89.7	59.0
Methods	Sky	Ped	Rid	Car	Tru	Bus	Tra	Mot	Bic	mIoU(%)
ERFNet [12]	93.1	73.0	53.2	91.8	59.1	70.1	66.7	44.9	67.1	70.0
LERNet(Ours)	92.1	70.3	50.2	91.9	67.2	75.3	69.5	47.2	66.9	69.5

Table 7
Comparison with some state-of-the-art methods on CamVid test set for single frame prediction.

Methods	Mean IoU(%)	Sky	Building	Pole	Road	Sidewalk	Tree	Sign	Fence	Car	Pedestrian	Bicyclist
SegNet [42]	55.6	92.4	88.8	27.5	97.2	84.4	87.3	20.5	49.3	82.1	57.1	30.7
ENet [10]	51.3	95.1	74.7	35.4	95.1	86.7	77.8	51.0	51.7	82.4	67.2	34.1
LERNet(Ours)	58.2	88.4	79.6	23.8	92.0	79.6	72.6	43.6	44.5	80.2	43.5	51.6

shows, the beginning column lists existing state-of-the-art semantic segmentation methods including PSPNet [40], DeepLab [41], SegNet [42], ENet [10], ESPNet [11], ERFNet [12], BiSeNet [14] and ICNet [13]. Althogh the first two methods predict with high pixel-level accuracy, they are not real-time methods. Each row of the table contains the performance of GFLOPs, parameter amount, inference speed, mIoU on CityScapes test set, mIoU on CityScapes validation set, input size, and GPU version. As can be seen in Table 5, our method can balance the performance of diverse metrics well. The LERNet can effectively reduce the FLOPs and the number of parameters, only requires 12.7 GFLOPs for 512 × 1024 inputs, and only has 0.65M parameters, which is one-ninth of the number of parameters of BiSeNet. Besides, our model is fast while maintaining high accuracy, achieves 100 fps, and 69.5% mIoU on the validation set. Performance of mIoU on the test set is 66.5%, which is worse than expected may be because we directly upsample our predicted results (512 × 1024) 2 times to match the original resolution (1024 × 2048). Note that when the input scale is enlarged to 768 × 1536, our LERNet requires 28.7 GFLOPs and achieves 90.9 fps, which is hardly changed. The performance of mIoU decreases by 4% after increasing the input scale, we speculate the reason is that the amount of parameters is too small to fully learn the features of the large-scale input image. Tables 6 and 7 separately display per-class performance of our model and the other state-of-the-art models on CityScapes validation set and Camvid test set. In Table 6, we can see that our LERNet can better predict transportation like car, truck, bus, train and motorbike. As displayed in Table 7, our LERNet achieves 58.2% mIoU on the CamVid test dataset.

For a more intuitive display, we draw Fig. 5 for various methods. The horizontal axis represents GFLOPs, and the vertical axis represents mIoU (%). Each method is a circle, the bigger the circle, the faster the inference speed of the method. As Fig. 5 illustrates, our method can successfully balance the performance of computational cost, accuracy and speed.

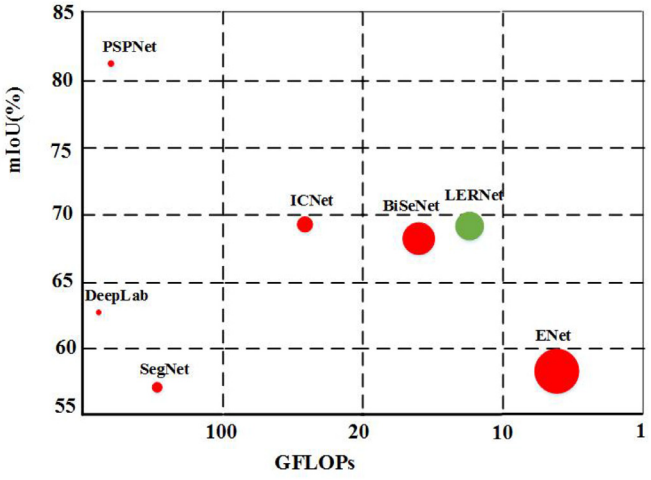

Fig. 5. MIoU, GFLOPs and inference speed performance on Cityscapes for single frame prediction. The radius of each circle indicates the inference speed, and the bigger the circle, the faster the speed. Results of PSPNet [40], DeepLab [41], SegNet [42], ENet [10], ESPNet [11], ERFNet [12], BiSeNet [14], ICNet [13] and our LERNet are shown. Note that the green circle is ours. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4.4. Experiments on the video semantic segmentation task

In this section, we carry out various experiments to prove the effectiveness and efficiency of our proposed method. Firstly,

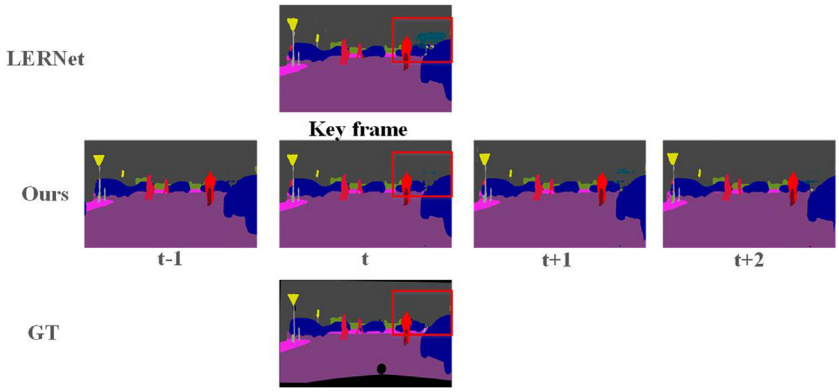

Fig. 6. Effectiveness of our THA module. From top to bottom: single frame segmentation result of our LERNet, continuous multi-frame segmentation results using THA module, and the ground-truth. Red boxes indicate where better segmentation result is obtained using our THA module.

Table 8
Ablation experiment on CityScapes validation set of the entire video semantic segmentation network.

	Time(ms)	Frames(fps)	mIoU(%)
Baseline	10	100	69.5
THA module	1	–	–
Non-key frames only	7	143	–
Video frames prediction (K=5)	7.6	131	60.6

Table 9
Comparison with some state-of-the-art methods for video semantic segmentation task on CityScapes dataset.

Method	mIoU(%)	Avg Time(ms)	Frames(fps)
DFF [6]	70.1	273	3.7
ClockNet [23]	67.7	141	7.1
GRFP(Dilation10, FlowNet2) [5]	69.5	685	1.4
Low-latency [37]	75.9	119	8.4
Ours	60.6	7.6	131.6

we analyse the effect of our feature propagation module (THA module). Then we compare our method with other state-of-the-art methods in the area of video semantic segmentation task on both CityScapes and Camvid. Besides, the prediction results on two datasets are visualized respectively.

4.4.1. Effect of THA module

In order to reduce information redundancy between adjacent video frames and further increase the speed while maintaining the accuracy of our model, we propose THA module to propagate features between key frames and non-key frames. To verify the implementation of the above objectives, we take our LERNet as baseline and perform ablation experiment on CityScapes validation set. As shown in Table 8, Time represents average time estimated on 100 frames with scale of 512 $\times$ 1024. THA module row shows that our THA module only needs 1ms to achieve feature propagation. Since the runtime for a key frame equals to that of our backbone network LERNet, Non-key frames only represents that we only measure the speed of processing non-key frames, which can be realized by feeding key frame and non-key frame pairs to our model. We extract low-level features of the key frame and the non-key frame. For non-key frames, we utilize the feature propagation network rather than the whole backbone network. Video frames prediction(K=5) indicates that we measure speed under the fixed key frame selection scheduling, and there is one key frame out of 5 consecutive frames. In Table 8, we can see that our THA module efficiently improves the speed of video frames prediction, increasing by 31fps while decreasing the performance of mIoU by 8.9%. Although the score drops, we find that using our THA module can effectively avoid some incorrect segmentation based on the consistency of consecutive frames. As shown in Fig. 6, from top to bottom are single frame segmentation result of our LERNet, continuous multi-frame segmentation results using THA module and the ground-truth. The middle column including segmentation results of the same key frame. The red box of each row points where to notice, we can see that our video-based method produces better result than our image-based baseline. Hence, we speculate that using THA module to propagate features can maintain across-frame consistency and avoid wrong predictions to some extent.

4.4.2. Comparison with the state-of-the-art methods for video semantic segmentation task

As far as we know, there are few real-time methods in the area of video semantic segmentation task. To prove the efficiency of our method, we carry out comparison experiment with other state-of-the-art video semantic segmentation methods. As shown in Table 9, we compare our method with DFF [6], ClockNet [23], GRFP [5] and Low-latency [37]. All of these methods are designed for video semantic segmentation task, and they also focus on the idea of feature propagation. For fair comparison, we use the fixed rate key frame selection scheduling, and choose a key frame every 5 frames. The DFF propagates high-level features from the key frame to current frame by optical flow learned in a separate flow network. The GRFP combines FCN and a gated recurrent unit(GRU) layer that is able to temporally propagate labeling information by using optical flow. ClockNet adapts multi-stages FCN and directly reuses the second or third stage features of previous frames to save computation. Besides, Low-latency designs adaptive key frame selection scheduling and propagate features efficiently by its adaptive propagation module. In Table 9, we can see that although our performance of accuracy is lower than the state-of-the-art methods, the inference speed of our model is much higher than these methods and already achieves the real-time level. Compared with ClockNet, its mIoU is 7.1% higher than ours, but we greatly increase the inference speed from 7.1fps to 131.6fps. To sum up, our model efficiently improves the inference speed and ensures that the accuracy drops within an acceptable range.

Performance on CamVid can be seen in Table 10. We lists several state-of-the-art methods in the beginning column, including SuperParsing [43], RTDF [44] and DAG-RNN [45]. It can be seen that the Pixel Accuracy (PA) of our model is 67.9%, which is 16% less than SuperParsing. Since the inference speed is not given in these works, we display ours in the right two columns. Our

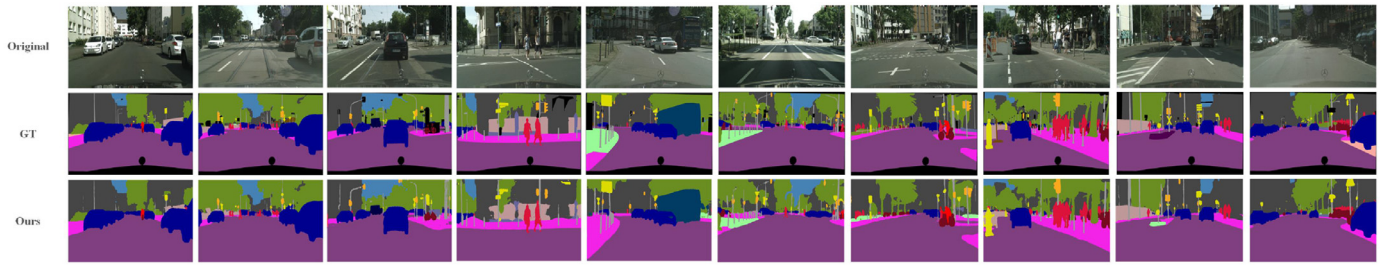

Fig. 7. Qualitative experiment on the CityScapes validation set for video semantic segmentation task.

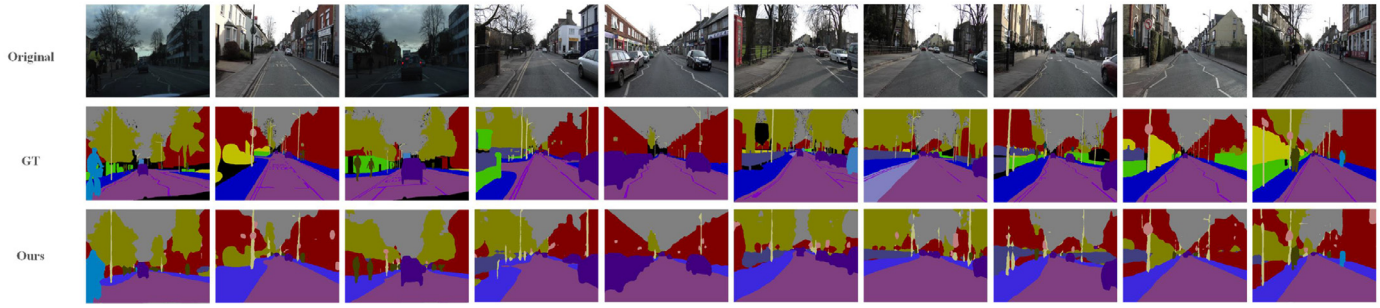

Fig. 8. Qualitative experiment on the CamVid test dataset for video semantic segmentation task.

Table 10
Comparison with some state-of-the-art methods for video semantic segmentation task on CamVid dataset.

Method	PA(%)	Avg Time(ms)	Frames(fps)
SuperParsing [43]	83.9	–	–
RTDF [44]	89.9	–	–
DAG-RNN [45]	91.6	–	–
Ours	67.9	5.8	172

model merely needs 5.8ms and achieves 172fps with the scale of 360 × 480. Note that our training set is not identical to that of [43–45]. Specifically, we do not have the big dataset with about 40K consecutive video frames, only 688 frames labelled at 30Hz are included in the training set. Hence, the comparison to other works is not strictly fair. Besides, we visualize the prediction results of our model on both CityScapes and Camvid, which can be viewed in Figs. 7 and 8. Note that pictures of the "GT" row in Figs. 7 and 8 have more colors than our predicted results in the last row, which is because that the number of classes needs to be predicted is less than that of the ground-truth.

5. Conclusion

In order to reduce information redundancy between adjacent video frames and improve the inference speed, we design a novel methodology to fully leverage across-frame consistency information and propagate features. Concretely, we first propose a real-time backbone network (LERNet) to extract features for a single frame. To further improve inference speed and reduce inter-frame redundancy, then we propose a Temporal Holistic Attention module (THA module) to propagate features. The attention weight matrix obtained by THA module measures similarities between non-key frame and key frame, and we apply it to high-level features of key frame to generate the predicted high-level features of non-key frame. In this way, we manage to realize feature propagation between two frames. Moreover, since attention weights represent degrees of correlation between two frames, the weighted features of key frame can successfully maintain inter-frame consistency. After that, to complement the unique features in non-key frames, we fuse the predicted high-level features with the original low-level features of the non-key frame. We experiment on two challenging datasets, i.e. CityScapes and CamVid, and compare the results with the existing state-of-the-art methods, which confirms that our approach realizes remarkable speedup while guaranteeing the accuracy. Since our method achieves real-time inference speed, it can be widely applied in areas like autonomous driving and robotics. In pursuit of high speed and low computational cost, our feature propagation module greatly compress the number of channels of the generated attention map, thus losing some channel information, which can be seen as a weakness of our model. For future work, we intend to exploit how to propagate features more efficiently and how to further improve video segmentation accuracy with lightweight network.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] W. Wang, Z. Zhang, S. Qi, J. Shen, Y. Pang, L. Shao, Learning compositional neural information fusion for human parsing, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 5703–5713.
[2] J. Ji, S. Buch, A. Soto, J. Carlos Niebles, End-to-end joint semantic segmentation of actors and actions in video, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 702–717.
[3] H. Yang, C. Yuan, B. Li, Y. Du, J. Xing, W. Hu, S.J. Maybank, Asymmetric 3d convolutional neural networks for action recognition, Pattern Recognit. 85 (2019) 1–12.
[4] H. Song, W. Wang, S. Zhao, J. Shen, K.-M. Lam, Pyramid dilated deeper ConvLSTM for video salient object detection, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 715–731.
[5] D. Nilsson, C. Sminchisescu, Semantic video segmentation by gated recurrent flow propagation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6819–6828.
[6] X. Zhu, Y. Xiong, J. Dai, L. Yuan, Y. Wei, Deep feature flow for video recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2349–2358.
[7] Y. Wang, J. Liu, Y. Li, J. Fu, M. Xu, H. Lu, Hierarchically supervised deconvolutional network for semantic video segmentation, Pattern Recognit. 64 (1) (2017) 437–445.

[8] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The cityscapes dataset for semantic urban scene understanding, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3213–3223.
[9] G.J. Brostow, J. Fauqueur, R. Cipolla, Semantic object classes in video: a high-definition ground truth database, Pattern Recognit. Lett. 30 (2) (2009) 88–97.
[10] A. Paszke, A. Chaurasia, S. Kim, E. Culurciello. ENet: a deep neural network architecture for real-time semantic segmentation, arXiv:1606.02147 (2017).
[11] S. Mehta, M. Rastegari, A. Caspi, L.G. Shapiro, H. Hajishirzi, ESPNet: efficient spatial pyramid of dilated convolutions for semantic segmentation, European Conference on Computer Vision (2018) 561–580.
[12] E. Romera, J.M. Alvarez, L.M. Bergasa, R. Arroyo, Erfnet: efficient residual factorized convnet for real-time semantic segmentation, IEEE Trans. Intell. Transp. Syst. 19 (1) (2018) 263–272.
[13] H. Zhao, X. Qi, X. Shen, J. Shi, J. Jia, ICNet for real-time semantic segmentation on high-resolution images, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 405–420.
[14] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, N. Sang, BiSeNet: bilateral segmentation network for real-time semantic segmentation, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 325–341.
[15] Y. Wang, Q. Zhou, J. Liu, J. Xiong, G. Gao, X. Wu, L.J. Latecki, LEDNet: a lightweight encoder-decoder network for real-time semantic segmentation, Int. Conf. Image Process. (2019) 1860–1864.
[16] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.
[17] W. Wang, J. Shen, R. Yang, F. Porikli, Saliency-aware video object segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 40 (1) (2017) 20–33.
[18] W. Wang, J. Shen, L. Shao, Video salient object detection via fully convolutional networks, IEEE Trans. Image Process. 27 (1) (2017) 38–49.
[19] W. Wang, J. Shen, J. Xie, M.-M. Cheng, H. Ling, A. Borji, Revisiting video saliency prediction in the deep learning era, IEEE Trans. Pattern Anal. Mach. Intell. (2019).
[20] P. Lei, S. Todorovic, Recurrent temporal deep field for semantic video labeling, in: European Conference on Computer Vision, Springer, 2016, pp. 302–317.
[21] W. Wang, Q. Lai, H. Fu, J. Shen, H. Ling, Salient object detection in the deep learning era: an in-depth survey, arXiv:1904.09146 (2019).
[22] W. Wang, J. Shen, L. Shao, Consistent video saliency using local gradient flow optimization and global refinement, IEEE Trans. Image Process. 24 (11) (2015) 4185–4196.
[23] E. Shelhamer, K. Rakelly, J. Hoffman, T. Darrell, Clockwork convnets for video semantic segmentation, in: European Conference on Computer Vision, Springer, 2016, pp. 852–868.
[24] Y. Li, J. Shi, D. Lin, Low-latency video semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 5997–6005.
[25] W. Wang, X. Lu, J. Shen, D.J. Crandall, L. Shao, Zero-shot video object segmentation via attentive graph neural networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 9236–9245.
[26] W. Wang, J. Shen, F. Porikli, R. Yang, Semi-supervised video object segmentation with super-trajectories, IEEE Trans. Pattern Anal. Mach. Intell. 41 (4) (2018) 985–998.
[27] X. Jin, X. Li, H. Xiao, X. Shen, Z. Lin, J. Yang, Y. Chen, J. Dong, L. Liu, Z. Jie, et al., Video scene parsing with predictive feature learning, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5580–5588.
[28] W. Wang, J. Shen, Deep visual attention prediction, IEEE Trans. Image Process. 27 (5) (2017) 2368–2378.
[29] W. Wang, J. Shen, X. Dong, A. Borji, R. Yang, Inferring salient objects from human fixations, IEEE Trans. Pattern Anal. Mach. Intell. (2019).
[30] W. Wang, H. Song, S. Zhao, J. Shen, S. Zhao, S.C. Hoi, H. Ling, Learning unsupervised video object segmentation through visual attention, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 3064–3074.
[31] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7132–7141.
[32] X. Wang, R. Girshick, A. Gupta, K. He, Non-local neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7794–7803.
[33] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, W. Liu, CCNet: criss-cross attention for semantic segmentation, in: International Conference on Computer Vision, 2019, pp. 603–612.
[34] J. Fu, J. Liu, H. Tian, Z. Fang, H. Lu, Dual attention network for scene segmentation, Comput. Vis. Pattern Recognit. (2019) 3146–3154.
[35] G. Li, I. Yun, J.H. Kim, J. Kim, DABNet: depth-wise asymmetric bottleneck for real-time semantic segmentation, British Machine Vision Conference (2019).
[36] M. Yang, K. Yu, C. Zhang, Z. Li, K. Yang, DenseASPP for semantic segmentation in street scenes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3684–3692.
[37] Y. Li, J. Shi, D. Lin, Low-latency video semantic segmentation, Comput. Vis. Pattern Recognit. (2018) 5997–6005.
[38] Y. Zhu, K. Sapra, F.A. Reda, K.J. Shih, S. Newsam, A. Tao, B. Catanzaro, Improving semantic segmentation via video propagation and label relaxation, Comput. Vis. Pattern Recognit. (2019) 8856–8865.
[39] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, Encoder-decoder with atrous separable convolution for semantic image segmentation, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 801–818.
[40] H. Zhao, J. Shi, X. Qi, X. Wang, J. Jia, Pyramid scene parsing network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2881–2890.
[41] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A.L. Yuille, DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, IEEE Trans. Pattern Anal. Mach. Intell. 40 (4) (2018) 834–848.
[42] V. Badrinarayanan, A. Kendall, R. Cipolla, SegNet: a deep convolutional encoder-decoder architecture for image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 39 (12) (2017) 2481–2495.
[43] J. Tighe, S. Lazebnik, Superparsing: scalable nonparametric image parsing with superpixels, Eur. Conf. Comput. Vis. (2010) 352–365.
[44] P. Lei, S. Todorovic, Recurrent temporal deep field for semantic video labeling, Eur. Conf. Comput. Vis. (2016) 302–317.
[45] B. Shuai, Z. Zuo, B. Wang, G. Wang, Dag-recurrent neural networks for scene labeling, Comput. Vis. Pattern Recognit. (2016) 3620–3629.

Junrong Wu received the B.E. degree from Central South University, Changsha, China, in 2018. She is currently pursuing the M.S. degree at Beijing Laboratory of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, Beijing, China. Her current research interests include image captioning, VQA, and semantic segmentation.

Zongzheng Wen received the B.E. degree from Hebei University of Technology, Tianjin, China, in 2018. He is now pursuing the M.S. degree at Beijing Laboratory of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, 100081, PR China. His research interests include deep learning, SLAM, image saliency detection and semantic segmentation.

Sanyuan Zhao received her Ph.D from Beijing Institute of Technology in 2012. She worked at School of Computer Science and Technology of Beijing Institute of Technology as a lecturer since 2012. Her research areas include computer vision, computer graphics, and virtual reality.

Kele Huang received the B.E. degree from Central South University, Changsha, China, in 2018. He is currently pursuing the M.S. degree at Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His current research interests include system software, deep learning, image or video segmentation and understanding.