Semantic Segmentation in Compressed Videos

Ang Li*
University of Manitoba
Winnipeg, Canada
lia34513@myumanitoba.ca

Yiwei Lu*
University of Manitoba
Winnipeg, Canada
luy2@cs.umanitoba.ca

Yang Wang
University of Manitoba
Winnipeg, Canada
ywang@cs.umanitoba.ca

***Abstract*—Existing approaches for semantic segmentation in videos usually extract each frame as an RGB image, then apply standard image-based semantic segmentation models on each frame. This is time-consuming. In this paper, we tackle this problem by exploring the nature of video compression techniques. A compressed video contains three types of frames, I-frames, P-frames, and B-frames. I-frames are represented as regular images, P-frames are represented as motion vectors and residual errors, and B-frames are bidirectionally frames that can be regarded as a special case of a P frame. We propose a method that directly operates on I-frames (as RGB images) and P-frames (motion vectors and residual errors) in a video. Our proposed model uses a ConvLSTM model to capture the temporal information in the video required for producing the semantic segmentation on P-frames. Our experimental results show that our method performs much faster than other alternatives while achieveing similar performance in terms of accuracies.**

I. Introduction

Semantic segmentation in videos is of crucial importance for real-time application such as autonomous driving. Existing approaches usually operate on a frame-by-frame basis. They first extract each frame as a regular RGB image, then apply standard semantic segmentation models on this frame. These methods suffer from very high computational cost or low speed. Typically, our video is compressed to 15 to 30 frames per second (fps). However, according to the frame-by-frame model, it takes 0.17s to provide semantic segmentation on one frame. For example, if the video is played at 30 fps and the video length is 2 minutes, the frame-by-frame model will take 10 minutes to provide semantic segmentation for the video. As a result, they are not applicable to real-time semantic segmentation scenarios such as self-driving.

Existing frame-by-frame approaches ignore the fact that videos usually come in compressed format for transmission and storage. In this paper, we propose a semantic segmentation method that directly operates on compressed videos. Working directly with compressed videos provides several advantages. First of all, since we do not need to extract frames from a video, our method can be much faster. Secondly, the compressed video directly provides the motion information that RGB images do not have. As a result, our method can directly take advantage of this information and consider temporal information of a video clip.

* Equal Contribution

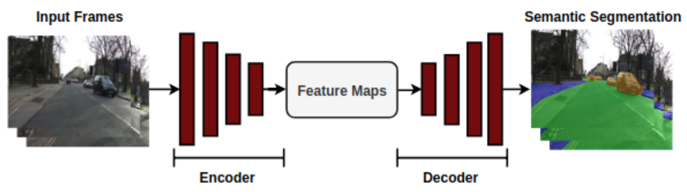

Fig. 1. Current solutions of semantic segmentation in videos require extracting all frames as regular RGB images, then process each image separately to produce its semantic segmentation. This can lead to heavy computation and low speed. In this paper, we propose a semantic segmentation method that directly operates on compressed videos without extracting all frames.

Existing works already explore the usability of compressed videos in computer vision tasks such as action recognition [18] and object detection [17]. However, to the best of our knowledge, this is the first work on using compressed videos in semantic segmentation. We propose a ConvLSTM model that propagates the temporal information from I-frame to succeeding P/B-frames for semantic segmentation. Our experimental results show that the proposed method performs either better or on-par with standard frame-based methods. But the proposed method can run at a much faster speed.

II. Related work

A. Semantic Segmentation

The goal of semantic segmentation is to assign a label to each pixel in an image (see Fig 1). For semantic segmentation in images, there have been a lot of models that apply deep convolutional neural networks [6], [7], [16] for semantic segmentations. For example, methods such as FCN [10], dilated convolutions [19] and SegNet [1] are widely used. To apply semantic segmentation on videos, the most popular approach is to extract each frame in the video as an image, then apply a standard image-based semantic segmentation algorithm to process each frame.

For semantic segmentation in videos, there is always a trade-off between accuracy and efficiency. In order to obtain higher accuracy, A new method for dealing with the spatial and temporal features of video semantic segmentation was proposed in [5]. A pyramid scene parsing network was applied in [20] to acquire more accurate semantic segmentation. But these methods requires a lot of computation time. A model that only focuses on a single annotation object was proposed in [3]. In order to reduce the computation time, a method based

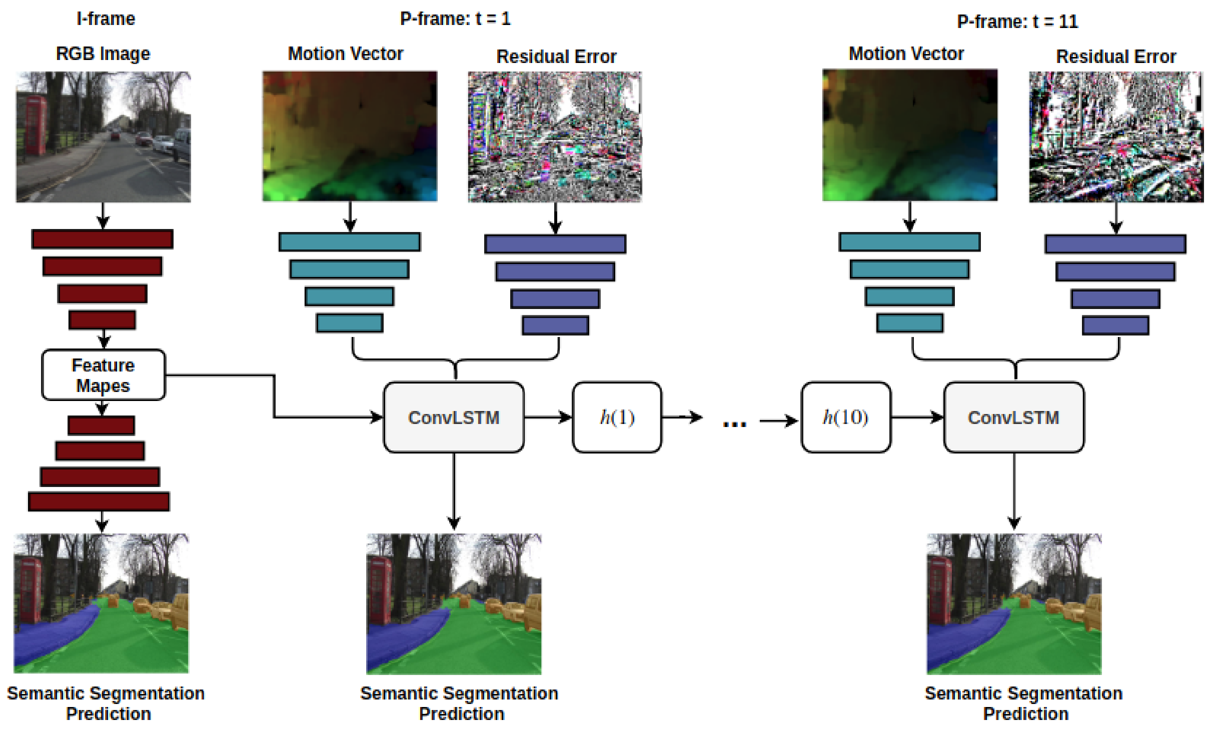

Fig. 2. We divide the video by groups. Each group contains 1 RGB image of the I-frame and 11 P-frames represented by the motion vector and the residual error. The processing of I-frames and P-frames are different: we first obtained a semantic segmentation of the I-frame based on ResNet. Then the information of the I-frame is taken as the initial state of a ConvLSTM module, which also takes the information of each P-frame to update its hidden state. At each time-step, the module produce a semantic segmentation prediction.

on clockwork driven by a fixed or adaptive clock signals was proposed in [9], [15].

B. Computer Vision with Compressed video

Videos usually come in compressed format for transmission and storage. Several popular compression techniques are widely applied, including AVI [12], MPEG4 [8], FLV [13], and so on. Recently, there has been some work on solving computer vision problems using compressed videos. For example, [18] uses MPEG4 on action recognition and their approach shows that operating on motion vectors and residual errors in compressed videos is more efficient than traditional methods that operate on RGB frames. [17] combines compressed video technology with LSTM to obtain spatial and temporal information on object detection problem. However, as far as we know, there is no existing work on semantic segmentation in compressed videos.

III. Approach

A. Overview

Videos are usually stored and transmitted in some compressed format, such as MPEG-4, H.264, etc. Most of the video compression techniques use the fact that adjacent frames in a video are often similar. As a result, we only need to store a small number of frames (called *I-frame*) as regular images, while other frames (called *P-frame*) can be efficiently represented by only storing the difference between frames.

Following prior work [18], we divide frames in an entire video into several groups, while each group contains one I-frame and several P-frames, represented by the collection $\{I, P_1, P_2, ... , P_T\}$. The I-frame I is represented as a regular RGB image, while each P-frame P_t only stores the difference with respect to the previous frame. Our model takes $\{I, P_1, P_2, ... , P_T\}$ as the input. The desired output is the semantic segmentation of each image, regardless of the frame type. The semantic segmentation network is represented by $f_s(x)$, where x can be either a I-frame or a P-frame. Given the ground-truth semantic segmentation masks, our learning objective function can be described below:

$$L = L_{ce}(GT_I - f_s(I)) + \sum_{t=1}^{T} L_{ce}(GT_{P_t} - f_s(P_t)) \qquad (1)$$

where L_{ce} is cross-entropy loss function, GT_I is the ground-truth semantic segmentation mask of the I-frame, and GT_{P_t} is the ground-truth semantic segmentation mask of the P-frame P_t. Our goal is to learn a network that minimizes the loss function defined in Eq. 1.

B. Semantic Segmentation for I-frame

In order to obtain the semantic segmentation of an I-frame, we use a standard encoder-decoder architecture for semantic segmentation (see Fig 2). An I-frame is represented as a regular RGB image tensor with three channels. Let

$I \in \mathbb{R}^{H \times W \times 3}$ be the image of the I-frame, where $H \times W$ is the spatial size of the image. We use ResNet as the backbone network to extract a feature map of the image denoted as $z(I) \in \mathbb{R}^{\frac{H}{32} \times \frac{W}{32} \times c}$, where c is the number of channels of the last convolutional layer of the feature extractor. We set c as the number of classes in semantic segmentation. The spatial size of $z(I)$ is smaller than the original image I due to max-pooling. In order to obtain the pixel-wise prediction at the original image size, we apply an upsampling layer to enlarge $z(I)$ to have the same spatial size of the input image. We use $f_s(I) \in \mathbb{R}^{H \times W \times c}$ to denote the output of this upsampling layer. We can interpret the c-dimensional vector at each pixel location of $f_s(I)$ as the score of classifying the pixel to each of the c classes.

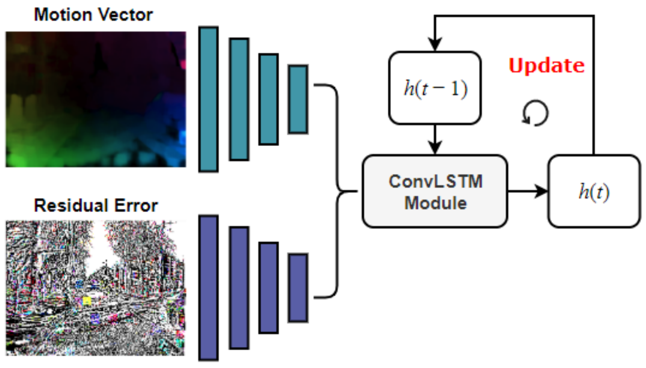

Fig. 3. The process of our network on P-frame when the timestep = t

C. Semantic Segmentation for P-frame

Since a P-frame is represened as the difference from the previous frame, an P-frame by itself does not contain enough information for semantic segmentation. In order to segment a P-frame, intuitively we should capture the temporal information between this P-frame and the preceding I-frame. In this work, we apply a ConvLSTM module to accumulate the information of previous frames (see Fig 3) that are needed for segmenting a particular P-frame at time t.

Let P_t denote the P-frame at time t. A P-frame is represented as the motion vector and the residual error (see Fig. 2). We can interpret the motion vector and the residual error as two images. We apply two different CNNs to extract features from these two images denoted as $z_1(t)$ and $z_2(t)$ (where $z_1, z_2 \in \mathbb{R}^{\frac{H}{32} \times \frac{W}{32} \times c}$), respectively. We then concatenate z_1 and z_2 as the input to ConvLSTM at time t (see Fig. 3).

The ConvLSTM module will process information starting from the I-frame in the group. We set the initial hidden state $h(0)$ of ConvLSTM as the feature of the corresponding I-frame, i.e. $h(0) = z(I)$. For the P-frame $P(t)$ at time t, we simply take the aforementioned concatenated features $cat(z_1(t), z_2(t))$ (where cat represents the concatenation operation) as the input at t in ConvLSTM.

We consider the hidden state $h(t)$ as the feature representation of the information of the P-frame $P(t)$. Since $h(t)$ has accumulated the information of frames starting from an I-frame that leads to $P(t)$, $h(t)$ has enough information for semantic segmentation of $P(t)$. We use $h(t)$ as the input and apply an upsampling layer to obtain the semantic segmentation.

IV. Experiments

In this section, we first describe our experimental setup and datasets in Section IV-A. We then present the experimental results in Section IV-B.

A. Experimental Setup

a) Datasets: We evalute the performance of our approach on the Cambridge-driving labelled video database (CamVid) dataset [2] and the Semantic Understanding of Urban Street Scenes dataset (Cityscapes) dataset [4]. CamVid provides object-class semantic labels that assign each pixel to one of the 32 semantic classes. Most videos are collected by using a fixed-position CCTV-style camera taken from the observation of a driver in the car. The driving scenes increase the number and heterogeneity of the observed object classes. We use three videos from CamVid: seq06R0, seq01TP, and seq 05VD. The total number of frames in the 1436 group is 17,239, and each group contains 12 frames (1 I-frame and 11 P-frames), with 19 semantic classes in the selected image. We divide the frame into 70% as training data (1005) and 30% (431) test data. Cityscapes provides an image segmentation dataset in an self-driving environment. It is used to evaluate the performance of visual algorithms in the semantic understanding of urban scenes. Cityscapes contains 50 different scenes, different backgrounds, different seasons of streetscapes. The total number of frames in the 960 group is 11,520 with 15fps and 19 class numbers. We divided the frame into 70% as training data (672) and 30% (288) test data.

b) Ground-truth labels: The videos in our evaluation datasets do not contain ground-truth labels for all frames. In order to get the ground-truth, we first decompress the video and extract all frames as regular RGB images. We then run ResNet [14] to obtain semantic segmentation maps for all frames based on their RGB images and use the predicted segmentation maps as the ground-truth.

c) Evaluation metrics: We use the mean Intersection Over Union (MeanIoU) and the pixel accuracy to measure the performance of the semantic segmentation. We also measure the speed of the proposed approach during inference time.

d) Baselines: We consider the following baseline methods for comparison. First, we consider standard semantic segmentation models that operate on regular images, including FCN-32s, FCN-8s, ResNet [5], [11]. Note that these baselines cannot directly handle compressed video format. They have to extract each frame as a regular image in order to predict the semantic segmentation of this frame. Since there is no existing work that directly produces semantic segmentation for compressed videos, we also define our own baseline as follows. This baseline first produces the semantic segmentation map on an I-frame. For remaining P-frames in the group, this baseline simply uses the semantic segmentation map from this

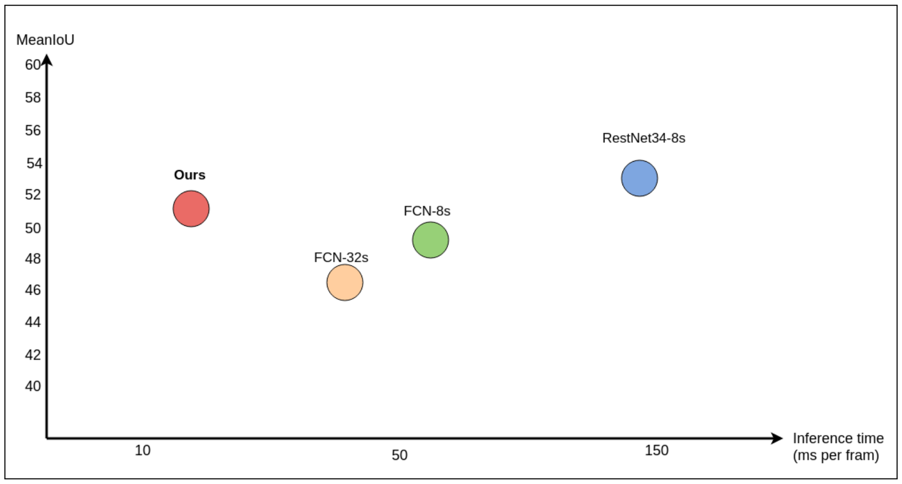

Fig. 4. Speed and accuracy on CamVid, compared to FCN-32s, FCN-8s, and ResNet.

Network	Pixel Accuracy	MeanIoU
FCN-32s [5]	91%	46.1%
FCN-8s [5]	92.6%	49.7%
ResNet [5]	95%	53%
Ours	**94%**	**51%**

TABLE I
EVALUATING PERFORMANCE BETWEEN FCN, RESNET, AND OURS APPROACH FOR VIDEO SEMANTIC SEGMENTATION ON CAMVID

Network	Inference time (ms per frame)
FCN-32s	42.5
FCN-8s	56
ResNet	168
Ours	**17**

TABLE II
EVALUATING INFERENCE TIME BETWEEN FCN, RESNET, AND OURS APPROACH FOR VIDEO SEMANTIC SEGMENTATION ON CAMVID

CamVid

Network	Pixel Accuracy	MeanIoU
Baseline	89%	25%
Ours	**94%**	**51%**

Cityscapes

Network	Pixel Accuracy	MeanIoU
Baseline	80%	22%
Ours	**87%**	**34%**

TABLE III
EVALUATING PERFORMANCE BETWEEN BASELINE AND OURS APPROACH FOR VIDEO SEMANTIC SEGMENTATION ON CAMVID AND CITYSCAPES

I-frame as the prediction for each P-frame.

B. Results

We first compare different methods in terms of both their accuracy and inference speed using the CamVid dataset. The comparisons are shown in Table I, Table II and Fig 4. Our method achieves better performance than FCN-32s and FCN-8s [5] in terms of both accuracy and speed. Our method performs comparably to ResNet in terms of MeanIoU and pixel accuracy, but our method is much faster.

We also compare the performance between our method and the baseline we have defined earlier in Table III. Our method achieves higher pixel accuracy and MeanIoU.

V. CONCLUSION

We have proposed a new method for semantic segmentation in compressed videos. Our method does not require extracting each frame as an RGB image. Instead, it directly operates on the compressed video format consisting of I-frames and P-frames. Our model uses a ConvLSTM model for capturing the temporal inforamtion required segmenting the P-frames. Our experimental results show that the proposed method performs on-par with frame-based methods in terms of accuracy. But our method can perform at a much higher speed during inference time. We believe our method can potentially be used in real-time applications where the efficiency is crucial.

VI. ACKNOWLEDGEMENT

This work was supported by NSERC. We thank NVIDIA for donating some of the GPUs used in this work.

REFERENCES

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[2] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *European Conference on Computer Vision*, 2008.

[3] S. Caelles, K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. V. Gool. One-shot video object segmentation. *CoRR*, 2016.

[4] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[5] M. Fayyaz, M. H. Saffar, M. Sabokrou, M. Fathy, F. Huang, and R. Klette. Stfcn: spatio-temporal fully convolutional neural network for semantic segmentation of street scenes. In *Asian Conference on Computer Vision*, 2016.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[8] D. J. LeGall. Mpeg (moving pictures expert group) video compression algorithm: a review. In *Image Processing Algorithms and Techniques II*, 1991.

[9] Y. Li, J. Shi, and D. Lin. Low-latency video semantic segmentation. *CoRR*, 2018.

[10] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[11] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[12] G. Maertens and K. Soroushian. Accurate and error resilient time stamping method and/or apparatus for the audio-video interleaved (avi) format, 2007.

[13] A. Mozo, M. Obien, C. Rigor, D. Rayel, K. Chua, and G. Tangonan. Video steganography using flash video (flv). In *IEEE Instrumentation and Measurement Technology Conference*, 2009.

[14] D. Pakhomov, V. Premachandran, M. Allan, M. Azizian, and N. Navab. Deep residual learning for instrument segmentation in robotic surgery. *arXiv preprint arXiv:1703.08580*, 2017.

[15] E. Shelhamer, K. Rakelly, J. Hoffman, and T. Darrell. Clockwork convnets for video semantic segmentation. *CoRR*, 2016.

[16] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[17] S. Wang, H. Lu, P. Dmitriev, and Z. Deng. Fast object detection in compressed video. *CoRR*, 2018.

[18] C.-Y. Wu, M. Zaheer, H. Hu, R. Manmatha, A. J. Smola, and P. Krähenbühl. Compressed video action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[19] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[20] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *IEEE conference on Computer Vision and Pattern Recognition*, 2017.