This WACV 2021 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

S3-Net: A Fast and Lightweight Video Scene Understanding Network by Single-shot Segmentation

Yuan Cheng[*‡] Yuchao Yang[†] Hai-Bao Chen[*] Ngai Wong[‡] Hao Yu[†]
[*]Shanghai Jiao Tong University, China
[†]Southern University of Science and Technology, China
[‡]The University of Hong Kong, Hong Kong
[*]{cyuan328, haibaochen}@sjtu.edu.cn [†]{yangyc3, yuh3}@sustech.edu.cn [‡]nwong@eee.hku.hk

Abstract

Real-time understanding in video is crucial in various AI applications such as autonomous driving. This work presents a fast single-shot segmentation *strategy for video scene understanding. The proposed net, called S3-Net, quickly locates and segments* target sub-scenes*, meanwhile extracts* structured time-series semantic features *as inputs to an LSTM-based spatio-temporal model. Utilizing tensorization and quantization techniques, S3-Net is intended to be lightweight for edge computing. Experiments using CityScapes, UCF11, HMDB51 and MOMENTS datasets demonstrate that the proposed S3-Net achieves an accuracy improvement of* 8.1% *versus the 3D-CNN based approach on UCF11, a storage reduction of* $6.9\times$ *and an inference speed of* 22.8 *FPS on CityScapes with a GTX1080Ti GPU.*

1. Introduction

Visual environment perception is critical for autonomous vehicles, say, in the advanced driver assistance system (ADAS), which requires real-time segmentation and understanding of driving scenes such as free-space areas and surrounding behaviors, etc. Compared to the solutions with LIDARs, RADARs, etc. [3, 16], the computer vision-based approaches with deep learning can adequately extract scene information [7, 2]. Nevertheless, these pixel-wise approaches are designed to segment all pixels in a frame, which incurs unnecessary computational complexity and low processing speed. Proposal-wise methods [15, 13] avoid handling all pixels by learning only the proposed object candidates, but still require multiple steps of computationally expensive candidate proposal methods. A large amount of segmentation time is wasted on the unadopted candidates or overlapped areas of candidates. Moreover, most existing methods do not consider the temporal relationship of objects (viz., activities) in video stream, which is practically essential for autonomous emergency-braking, forward-collision avoidance and behavior-anticipation systems. As there are numerous possible activities of pedestrians and vehicles in the real driving environment, it is challenging to perform fast video scene understanding using existing segmentation networks.

To overcome these hurdles, we design S3-Net (a scene understanding network by **S**ingle-**S**hot **S**egmentation) for real-time video analysis in autonomous driving. The contributions come from fourfold:

- We devise a *single-shot segmentation* strategy to quickly locate and segment the *target sub-scenes* (optimized object areas without background), instead of segmenting all pixels or every object candidate in a frame.
- We build an LSTM-based spatio-temporal model based on the *structured time-series semantic features* extracted from the former segmentation model for activity recognition in video stream.
- We realize both *object segmentation* and *activity recognition*, for the first time, in a single lightweight framework.
- We develop a *structured tensorization* of the LSTM-based spatio-temporal model, which results in *accuracy improvement even under deep compression* and hence can be used on terminal/edge devices.

Experimental results on CityScapes [9], UCF11 [21], HMDB51 [18] and MOMENTS [21] show that the proposed method achieves a remarkable accuracy improvement of 8.1% over the 3D-CNN based approach on UCF11, a storage reduction of $6.9\times$ and an inference speed of 22.8 FPS on CityScapes with a GTX1080Ti GPU.

In the following, Section 2 reviews the related works. Section 3 presents the proposed S3-Net. Section 4 introduces the further improvements of S3-Net by structured

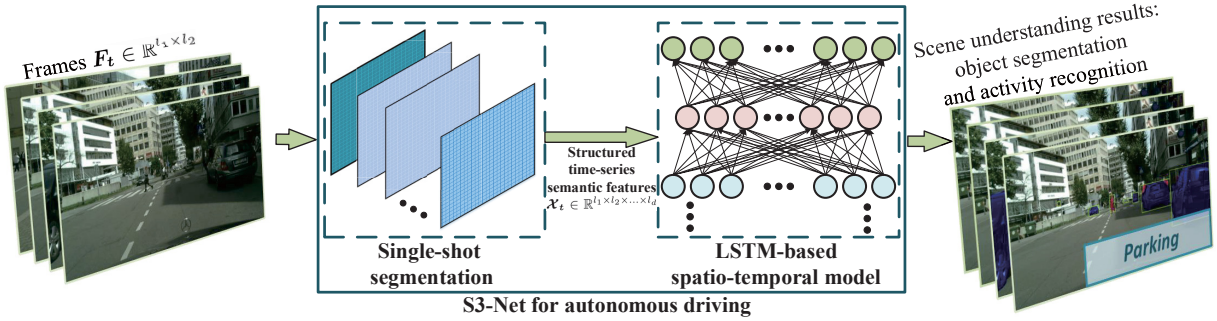

Figure 1. S3-Net: a single-shot segmentation network for fast video scene understanding towards autonomous driving.

tensorization and trained quantization. Section 5 provides the experimental results on several large-scale datasets, followed by conclusion in Section 6 concludes the paper.

2. Related Work

Modern researches on segmentation mainly fall into 3 categories.

Pixel-wise Existing pixel-wise approaches for segmentation are designed to predict a category label for *each pixel*, which are usually realized by fully convolutional networks (FCNs) [2].Various improvements like dilated convolutions [33] are further developed for enhanced performance. These methods are, however, limited with slow runtime and relatively low accuracy.

Proposal-wise Driven by the advancement of object detection networks, recent works perform *instance* segmentation with R-CNN to first propose object candidates and then segment all of them. The work in [10] utilizes the shared convolutional features among object candidates in segmentation layers. Multi-task cascaded network [11] is developed with an instance-aware semantic segmentation on object candidates. Mask R-CNN [15] is developed as the extension of Faster R-CNN with a mask branch. All these approaches require *multiple steps* that first generates object candidates, then segments all of them, and at last detects and recognizes the correct ones. Apparently, such object proposal methods waste unnecessary computation on the unadopted candidates and overlapped areas of candidates.

Single-stage Lately, there are attempts to produce a single-stage segmentation. FCIS [19] assembles the position-sensitive score maps within the ROI to directly predict segmentation results. YOLACT [4] tries to combine the prototype masks and predicted coefficients and then crops with a segmented bounding box. PolarMask [32] introduces the polar representation to formulate pixel-wise segmentation as a distance regression problem. SOLO [31] divides network into two branches to generate instance segmentation with predicted object locations. However, they still require significant amounts of pre- or post-processing before or after localization, and cannot achieve a real-time speed.

Moreover, in the real driving environment, vehicles require precise scene understanding not only segmentation. In contrast to all aforementioned approaches, we propose the practical scene understanding network S3-Net for autonomous driving. S3-Net adopts a single-shot segmentation model to quickly locate and segment the target sub-scenes; and an LSTM-based spatio-temporal model to precisely recognize activities from the structured time-series semantic features. With elaborated tensorization and quantization algorithms, the proposed framework provides a fast and lightweight scene understanding for vehicle-mounted edge/terminal devices.

3. S3-Net

This section elaborates the proposed scene understanding network S3-Net, as shown in Fig. 1. It leverages object segmentation and activity recognition, for the first time, in a single lightweight framework. Our design targets 3 criteria, namely, real-time speed, high accuracy and small size.

3.1. Single-shot Segmentation

To precisely detect the free-space areas and determine the following moves, the frames in autonomous driving are usually high-resolution (e.g., 2048×1024), which contain a huge number of pixels. We divide these pixels into 2 parts: **1)** Target object areas, which are important but practically minority in frames. **2)** Background areas, which are the majority in most situations. This implies significant processing time can be saved if target areas in a frame can be quickly and precisely located. With such analysis, we propose the single-shot segmentation strategy. Instead of handling all pixels (e.g., SegNet [2]) or every object candidate (e.g., Mask R-CNN [15]) in a frame, the single-shot segmentation focuses on *only* segmenting the target sub-scenes of *optimized* object areas without background, as shown in Fig. 2.

In the proposed single-shot segmentation, we regard the sub-scene detection as a single-shot regression problem and directly learn sub-scene coordinates and class probabilities from raw features. Assuming that $\boldsymbol{F_t} \in \mathbb{R}^{l_1 \times l_2}$ are the

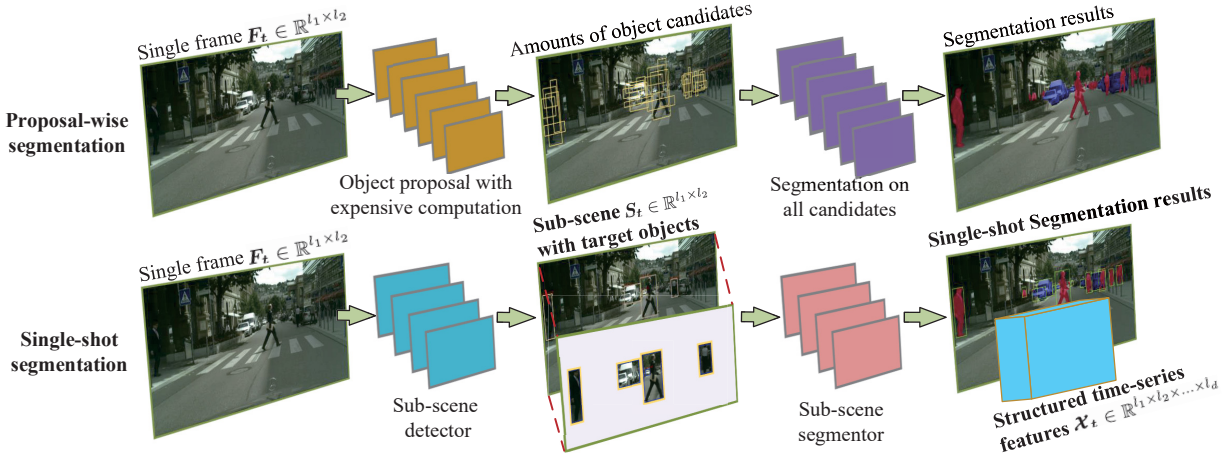

Figure 2. Comparison between the proposal-wise segmentation and the proposed single-shot segmentation.

video frames and $\boldsymbol{S_t} \in \mathbb{R}^{l_1 \times l_2}$ are the target sub-scenes of optimized object areas, where subscript t denotes the time sequence and l represents the mode size of dimension. First, the *sub-scene detector* is employed to locate target sub-scenes:

$$\boldsymbol{S_t} = detc(\boldsymbol{F_t}), \tag{1}$$

using $detc$ operation to represent the sub-scene detection processing. Note that we set the number of sub-scenes in a frame to be lower than a certain value (in our experiments is 25), and we skip the frame if no sub-scene detected. After obtaining the target sub-scenes, we apply the *sub-scene segmentor* with fewer layers than the proposal-wise methods to deliver even higher accuracy. Furthermore, as seen in Fig. 2, the semantic features are extracted from the last convolutional layer of single-shot segmentation model for activity recognition, which will be discussed next.

3.2. Spatio-temporal Model

In practice, we construct a spatio-temporal model based on an LSTM network using the structured time-series semantic features aggregated from each frame. Suppose that $\boldsymbol{\mathcal{X}_t} \in \mathbb{R}^{l_1 \times l_2 \times \ldots \times l_d}$ (here dimensions l_1, l_2, etc. are generic and not to be confused with those in $\boldsymbol{F_t}$ and $\boldsymbol{S_t}$) are the time-series semantic features structured into the tensor format, where d is the dimensionality of the tensor. The single-shot segmentation model uses several convolutional layers to learn structured time-series semantic features from frames:

$$\boldsymbol{\mathcal{X}_t} = extr(\boldsymbol{S_t}), \tag{2}$$

the $extr$ operation represents the corresponding extraction method in the proposed feature extractor. Specifically, the $\boldsymbol{\mathcal{X}_t}$ are *structured* as an $s \times f \times c$ tensor, where s is number of sub-scenes for each frame, f denotes the learned features for each sub-scene, and c represents confidence scores for those sub-scenes. Then the LSTM cells (consisting of fully-connected layers) in the LSTM network take $\boldsymbol{\mathcal{X}_t}$ as inputs, instead of direct video frames $\boldsymbol{F_t}$, to learn the spatio-temporal information. Each LSTM cell keeps track of an internal state that represents its memory and learns to update its state over time based on the current input and past states, as in the following:

$$\begin{aligned}
&\boldsymbol{\mathcal{E}_t} = \sigma(\boldsymbol{\mathcal{W}_e}\boldsymbol{\mathcal{X}_t} + \boldsymbol{\mathcal{U}_e}\boldsymbol{\mathcal{H}_{t-1}} + \boldsymbol{\mathcal{B}_e}), \boldsymbol{\mathcal{Z}_t} = \sigma(\boldsymbol{\mathcal{W}_z}\boldsymbol{\mathcal{X}_t} + \boldsymbol{\mathcal{U}_z}\boldsymbol{\mathcal{H}_{t-1}} + \boldsymbol{\mathcal{B}_z}),\\
&\boldsymbol{\mathcal{D}_t} = \sigma(\boldsymbol{\mathcal{W}_d}\boldsymbol{\mathcal{X}_t} + \boldsymbol{\mathcal{U}_d}\boldsymbol{\mathcal{H}_{t-1}} + \boldsymbol{\mathcal{B}_d}), \tilde{\boldsymbol{\mathcal{C}}}_t = tanh(\boldsymbol{\mathcal{W}_c}\boldsymbol{\mathcal{X}_t} + \boldsymbol{\mathcal{U}_c}\boldsymbol{\mathcal{H}_{t-1}} + \boldsymbol{\mathcal{B}_c}),\\
&\boldsymbol{\mathcal{C}_t} = \boldsymbol{\mathcal{E}_t} \odot \boldsymbol{\mathcal{C}_{t-1}} + \boldsymbol{\mathcal{Z}_t} \odot \tilde{\boldsymbol{\mathcal{C}}}_t, \boldsymbol{\mathcal{H}_t} = \boldsymbol{\mathcal{D}_t} \odot tanh(\boldsymbol{\mathcal{C}_t}),
\end{aligned} \tag{3}$$

where $\odot$ denotes the element-wise product, $\sigma(\circ)$ represents the sigmoid function and $tanh(\circ)$ represents the hyperbolic tangent function. $\boldsymbol{\mathcal{H}_{t-1}}$ and $\boldsymbol{\mathcal{C}_{t-1}}$ are the previous hidden state and previous update factor, $\boldsymbol{\mathcal{H}_t}$ and $\boldsymbol{\mathcal{C}_t}$ are the current hidden state and current update factor, respectively. The weight matrices $\boldsymbol{\mathcal{W}}$ and $\boldsymbol{\mathcal{U}}$ weigh the input $\boldsymbol{\mathcal{X}_t}$ and the previous hidden state $\boldsymbol{\mathcal{H}_{t-1}}$ to update factor $\tilde{\boldsymbol{\mathcal{C}}}_t$ and three sigmoid gates, namely, $\boldsymbol{\mathcal{E}_t}$, $\boldsymbol{\mathcal{Z}_t}$ and $\boldsymbol{\mathcal{D}_t}$. Note that all these data structures have been tensorized and quantized, which is further discussed in Section 4.

For each frame in autonomous driving, the spatio-temporal model calculates its information by combining previous and current features. Therefore, all temporal information in video stream can be captured from the beginning till the current frame, and then activities can be recognized. Note that we make use of *structured time-series semantic features* instead of the direct video frames as inputs to the LSTM, as shown in Fig. 2. This way, the LSTM is fed with structured and distilled sub-scene information yielding high accuracy and performance.

3.3. Video Scene Understanding

Based on the proposed single-shot segmentation and spatio-temporal models, S3-Net can run a fast object segmentation and activity recognition, whose workflow is shown in Fig. 3. First, the raw video frames are fed into the

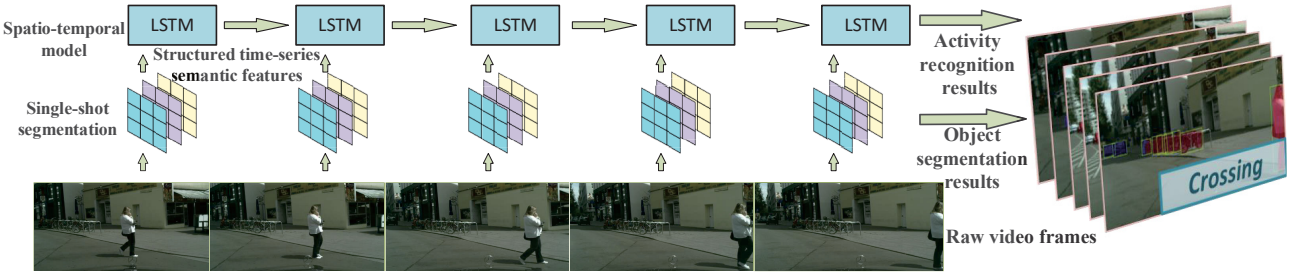

Figure 3. Workflow of S3-Net based scene understanding: object segmentation and activity recognition.

single-shot segmentation model, the object segmentation results and semantic features of each frame are stacked. Then, the structured time-series semantic features are fed into the spatio-temporal LSTM model. Finally, after processing the deeply learned features, activities are recognized. As a result, the proposed S3-Net represents a highly-optimized approach to autonomous driving.

4. Other Improvements

To deal with high-dimensional video-scale inputs, the weight matrix mapping from the input to the hidden layer becomes extremely large. To address this issue, we present the structured tensorization and trained quantization algorithms during the training of the S3-Net as follows.

4.1. Structured Tensorization

A tensor is a d-dimensional generalization of a vector or matrix, denoted by calligraphic letters $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{l_1 \times l_2 \times \ldots \times l_d}$ where $\boldsymbol{\mathcal{X}}(h_1, h_2, \ldots, h_d)$ is an element specified by the indices $h_1, h_2, \ldots, h_d$. One can tensorize a vector $\boldsymbol{x}$ or matrix $\boldsymbol{X}$ into a high-dimensional tensor $\boldsymbol{\mathcal{X}}$ using the $reshape$ operation, as depicted in Fig. 4. The total number of elements is $l_1 l_2 \cdots l_d$ which grows exponentially as d increases. In practice, tensor decomposition is used to find a low-rank approximation that expresses the original tensor by a number of small tensor factors. This often reduces the computational complexity from exponential to only linear, thereby eluding the *curse of dimensionality*.

In S3-Net, the initial inputs of spatio-temporal model are time-series semantic features, which are already *structured* as an $s \times f \times c$ tensor. In practice, we adopt a structured tensorization strategy to advance S3-Net. Given a d-dimensional feature tensor $\boldsymbol{\mathcal{X}}$, the tensorization reads

$$\boldsymbol{\mathcal{X}}(h_1, h_2, \ldots, h_d) = \sum_{\alpha_1, \ldots, \alpha_{d-1}}^{r_1, \ldots, r_{d-1}} \boldsymbol{\mathcal{G}_1}(1, h_1, \alpha_1) \boldsymbol{\mathcal{G}_2}(\alpha_1, h_2, \alpha_2) \ldots \boldsymbol{\mathcal{G}_d}(\alpha_{d-1}, h_d, 1), \tag{4}$$

where $\boldsymbol{\mathcal{G}_k} \in \mathbb{R}^{r_{k-1} \times l_k \times r_k}$ is the tensor core and r_k is the tensor train rank, α_k is the summation index ranging from 1 to r_k. Using the notation $\boldsymbol{\mathcal{G}_k}(h_k) \in \mathbb{R}^{r_{k-1} \times r_k}$ (a matrix slice from the 3-dimensional tensor $\boldsymbol{\mathcal{G}_k}$), (4) can be written

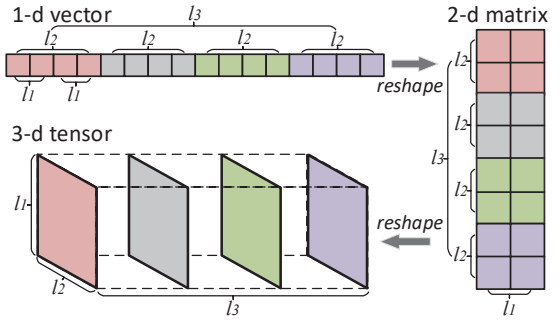

Figure 4. Reshaping a vector into a matrix and then into a 3-dimensional tensor.

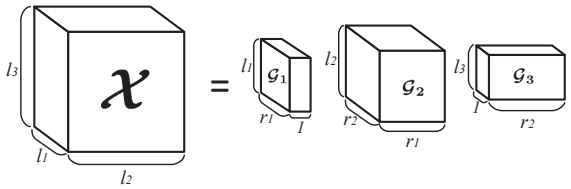

Figure 5. Tensor decomposition of a 3-dimensional tensor.

compactly as

$$\boldsymbol{\mathcal{X}}(h_1, h_2, \ldots, h_d) = \boldsymbol{\mathcal{G}_1}(h_1) \boldsymbol{\mathcal{G}_2}(h_2) \ldots \boldsymbol{\mathcal{G}_d}(h_d). \tag{5}$$

The decomposition of a 3-dimensional tensor is intuitively shown in Fig. 5. Since each integer l_k in (5) can be further decomposed as $l_k = n_k \cdot m_k$, each tensor core $\boldsymbol{\mathcal{G}_k}$ can be reformed with $\boldsymbol{\mathcal{G}_k^t} \in \mathbb{R}^{n_k \times m_k \times r_{k-1} \times r_k}$, and $\boldsymbol{\mathcal{G}_k^t}(j_k, i_k) \in \mathbb{R}^{r_{k-1} \times r_k}$. Therefore, the decomposition for the tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{(n_1 \times m_1) \times (n_2 \times m_2) \times \ldots \times (n_d \times m_d)}$ can be reformulated as:

$$\boldsymbol{\mathcal{X}}((j_1, i_1), (j_2, i_2), \ldots, (j_d, i_d)) = \boldsymbol{\mathcal{G}}_1^t(j_1, i_1) \boldsymbol{\mathcal{G}}_2^t(j_2, i_2) \ldots \boldsymbol{\mathcal{G}}_d^t(j_d, i_d). \tag{6}$$

Such double-index trick is then used to tensorize the LSTM-based spatio-temporal model in S3-Net, as shown in Fig. 6. Specifically, the most costly computation in LSTM is the large-scale matrix-vector multiplication generically represented as $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$ where $\boldsymbol{W} \in \mathbb{R}^{N \times M}$ is the weight matrix, $\boldsymbol{x} \in \mathbb{R}^M$ is the feature vector, $\boldsymbol{b} \in \mathbb{R}^N$ is the bias vector. To approximate $\boldsymbol{W}\boldsymbol{x}$ with much fewer parameters, we first reshape $\boldsymbol{W} \in \mathbb{R}^{N \times M}$ into a tensor $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{(n_1 \times n_2 \times \cdots \times n_d) \times (m_1 \times m_2 \times \cdots \times m_d)}$, where $N = \Pi_{k=1}^d n_k$ and $M = \Pi_{k=1}^d m_k$. Following (6), $\boldsymbol{\mathcal{W}}(h_1, h_2 \cdots, h_d)$ can

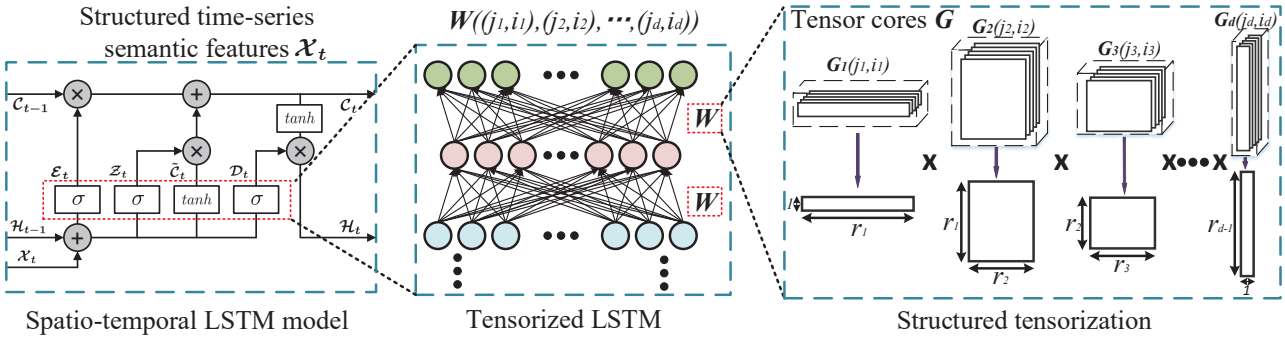

Figure 6. Structured tensorization of the spatio-temporal LSTM model.

be rewritten as $\mathcal{G}_1^t(j_1,i_1)\mathcal{G}_2^t(j_2,i_2)\ldots\mathcal{G}_d^t(j_d,i_d)$. Similarly, we can reshape $\boldsymbol{x}\in\mathbb{R}^M$, $\boldsymbol{b}\in\mathbb{R}^N$ into d-dimensional tensors $\boldsymbol{\mathcal{X}}\in\mathbb{R}^{m_1\times m_2\times\ldots\times m_d}$, $\boldsymbol{\mathcal{B}}\in\mathbb{R}^{n_1\times n_2\times\ldots\times n_d}$. As a result, the output $\boldsymbol{y}\in\mathbb{R}^N$ also becomes a d-dimensional tensor $\boldsymbol{\mathcal{Y}}\in\mathbb{R}^{n_1\times n_2\times\ldots\times n_d}$. Therefore, the matrix-vector multiplication can be expressed in the tensor form with usually low-rank cores

$$\boldsymbol{\mathcal{Y}}(j_1,j_2,\ldots,j_d)=\sum_{i_1=1}^{m_1}\sum_{i_2=1}^{m_2}\cdots\sum_{i_d=1}^{m_d}[\mathcal{G}_1^t(j_1,i_1)\mathcal{G}_2^t(j_2,i_2)\cdots \mathcal{G}_d^t(j_d,i_d)\boldsymbol{\mathcal{X}}(i_1,i_2,\ldots,i_d)]+\boldsymbol{\mathcal{B}}(j_1,j_2,\ldots,j_d). \tag{7}$$

The settings of m_k and n_k in our structured tensorization are determined by 2 criteria: **1)** Make the tensorization-based parameters uniformly small; **2)** Keep the sizes of dimensions not far from the already-structured inputs (in our experiments we structure $25\times425\times8$ into $25\times25\times17\times8$). This way, accuracy improvement can be maintained even under deep compression, which will be reported in Section 5.

4.2. Trained Quantization

The network processing with full-precision parameters requires unnecessarily large software and hardware resources. Here we present a quantization strategy on the whole S3-Net framework for further improvement. Note that we apply the quantized constraints during both network training and inference, called the *trained quantization*. Since the main parameters in S3-Net are weights and features, the trained quantization with 8-bit weights and features can result in high compression and efficiency. Note that such particular choice of 8-bit is determined by several S3-Net realizations from 4-bit to 10-bit. Assuming w_k is the full-precision weight entry, it can be quantized into its 8-bit counterpart w_k^q as:

$$w_k^q=\begin{cases}\frac{w_k}{|w_k|}, & 0<|w_k|\le\frac{1}{2^7},\\ floor(2^7\times w_k), & \frac{1}{2^7}<|w_k|<1,\\ (2^7-1)\frac{w_k}{|w_k|}, & |w_k|\ge1,\\ 0, & |w_k|=0,\end{cases} \tag{8}$$

where the function $floor$ takes the smaller nearest integer. We also enforce 8-bit features by quantizing a real feature element x_k into its 8-bit $x_k^q\in[0,1]$:

$$x_k^q=\frac{1}{2^8}\times\begin{cases}floor(2^8\times x_k), & 0\le x_k<1,\\ 2^8-1, & x_k\ge1.\end{cases} \tag{9}$$

Note that the batch normalization and max-pooling layers are also quantized into 8-bit similarly.

Based on proposed structured tensorization and trained quantization, we tensorize all matrix-vector products in the S3-Net similarly to (7) and quantize all tensor core entries (i.e. those entries in $\boldsymbol{\mathcal{G}_1},\cdots,\boldsymbol{\mathcal{G}_d}$) into 8-bit. Due to these improvements, the computational complexity of S3-Net reduces from $O(n_m^d)$ to $O(dr_{max}^2 n_m)$, where r_{max} is the maximum rank of cores $\boldsymbol{\mathcal{G}}_k$, and n_m is the maximum model size $n_k\cdot m_k$ of tensor weights $\boldsymbol{\mathcal{W}}$.

5. Experiments

The advantages of the S3-Net are demonstrated by comparisons with state-of-the-art results. Our experimental setup employs Tensorflow for coding and NVIDIA GTX-1080Ti for hardware realization. We validate S3-Net by evaluations on 1 large-scale segmentation dataset: CityScapes [9] and 3 challenging activity recognition datasets: UCF11 [21], HMDB51 [18] and MOMENTS [25].

5.1. Evaluation on Object Segmentation

To verify the performance of S3-Net on video object segmentation, we apply the CityScapes for comparison. This large-scale dataset contains high-quality pixel-level annotations of 5000 images of 2048×1024 resolution collected in street scenes from 50 different cities. Following the evaluation protocol for the single-shot segmentation and further activity recognition, we select 8 object labels: $person$, $rider$, car, $truck$, bus, $train$, $motorcycle$, $bicycle$ (belonging to 2 super categories: $human$ and $vehicle$), which have the possibility of performing an activity, and all other labels are considered as background. Note that the sub-scene detector has been pre-trained on COCO [20] with these 8 categories. The training, validation, and test sets contain 2975, 500 and 1525 images, respectively.

Figure 7. Sample visual results of S3-Net on CityScapes.

Approach	AP	AP_{50}	*person*	*rider*	*car*	*truck*	*bus*	*train*	*motorcycle*	*bicycle*
Pixel-level-Encoding [27]	8.9	21.1	-	-	-	-	-	-	-	-
InstanceCut [17]	13.0	27.9	10.0	8.0	23.7	14.0	19.5	15.2	9.3	4.7
SGN [22]	25.0	44.9	21.8	20.1	39.4	24.8	33.2	30.8	17.7	12.4
PolygonRNN++ [1]	27.6	44.6	-	-	-	-	-	-	-	-
SegNet [2]	29.5	55.6	29.9	23.4	43.4	29.8	41.0	**33.3**	18.7	16.7
SSAP [12]	**32.7**	51.8	35.4	25.5	**55.9**	**33.2**	**43.9**	31.9	19.5	16.2
Mask R-CNN [15]	26.2	49.9	30.5	23.7	46.9	22.8	32.2	18.6	19.1	16.0
Mask R-CNN[COCO] [15]	32.0	58.1	34.8	27.0	49.1	30.1	40.9	30.9	24.1	18.7
PA-Net [23]	31.8	57.1	**36.8**	**30.4**	54.8	27.0	36.3	25.5	22.6	**20.8**
GMIS [24]	27.3	45.6	31.5	25.2	42.3	21.8	37.2	28.9	18.8	12.8
Box2Pix [28]	13.1	27.2	-	-	-	-	-	-	-	-
S3-Net	32.3	**57.2**	35.8	27.9	51.3	29.7	39.5	29.1	**24.3**	20.4

"-" represents not reported or no open source for evaluation.

Table 1. Accuracy comparison with state-of-the-arts on CityScapes.

The segmentation accuracy is measured in terms of the standard average precision metrics: AP and AP_{50}, where AP_{50} represents the score over intersection-over-union (IoU) threshold 0.5. Moreover, the individual AP scores for every class are further evaluated. Some state-of-the-art results on CityScapes are chosen for accuracy comparison, as listed in Table 1. It can be seen in Table 1 that S3-Net outperforms various approaches and is only slightly lower than SSAP [12]. Specifically, the AP of S3-Net reaches 32.3, which is 0.3 higher than the Mask R-CNN[COCO] [15] and 0.5 higher than the PA-Net [23]. Sample visual results on CityScapes are presented in Fig. 7. It is found that S3-Net can precisely locate and segment the target sub-scenes, even for crowds in the distance.

5.2. Evaluation on Activity Recognition

For activity recognition, we use UCF11 and HMDB51 video datasets for accuracy comparison. The UCF11 contains 1600 video clips, falling into 11 activity classes that summarize the human activities visible in each clip such as *biking*, *diving* or *walking*. We resize the RGB frames into 160×120 at the FPS of 24 and sample all frames of each video clip as the input data. The HMDB51 provides 3 train-test splits each consisting of 5100 videos, falling into 51 classes of human activities like *Drink*, *Jump* or *Throw*. The training set contains 3570 videos (70 per class) and the test set has 1530 videos (30 per class). Each video has an FPS of 30. Table 2 shows the comparison between S3-Net with state-of-the-art results on the UCF11 and HMDB51 datasets. It can be seen that S3-Net significantly outperforms other approaches. Specifically, on UCF11 dataset, the top-1 accuracy of S3-Net reaches 98.3%, 8.1% higher

Approach	UCF11	HMDB51
Bag-of-words approach [21]	71.2%	59.4%
Two-stream CNN [26]	73.3%	66.4%
Original LSTM [14]	76.1%	69.6%
CNN+RNN [29]	83.7%	67.6%
3D-CNN [6]	89.2%	78.6%
Tensorized LSTM [8]	93.2%	79.5%
Temporal Segment Networks [30]	94.2%	69.4%
Two-Stream I3D [5]	97.9%	80.2%
S3-Net	**98.3%**	**80.8%**

Table 2. The activity recognition accuracy (top-1) comparison on UCF11 and HMDB51 datasets.

Figure 8. Sample visual results of S3-Net based scene understanding on MOMENTS.

Task	Approach	Storage(MB)	FPS(CityScapes)	FPS(MOMENTS)
Object segmentation	SegNet [2]	112	2.4	15.7
	SSAP [12]	-	3.4	19.2
	Mask R-CNN [15]	245.6	6.9	41.5
	PA-Net [23]	245.6	5.3	34.7
	Box2Pix [28]	-	10.9	-
Activity recognition	Two-stream CNN [26]	243.2	3.3	20.1
	Original LSTM [14]	616.3	5.9	38.0
	CNN+RNN [29]	720.5	-	11.5
	3D-CNN [6]	395.7	8.2	48.3
Object segmentation + Activity recognition	S3-Net	**89.2**	**22.8**	**137.3**

Table 3. The model size and speed comparisons on CityScapes and MOMENTS.

than the 3D-CNN [6] and 4.1% higher than the Temporal Segment Networks [30]. The quantitative comparison results demonstrate the *unique* benefit of the proposed S3-Net arises from the use of structured tensorization, namely, *accuracy improvement even under deep compression*.

We further report experimental results on the large-scale video dataset MOMENTS that contains one million labeled 3-second video clips involving people, animals, objects and natural phenomena that capture the gist of a dynamic scene. Each clip is assigned with 339 activity classes such as $walking$, $playing$ or $jogging$. Based on the majority of the clips, we resize every frame to a standard size of 340×256 at an FPS of 25. After training, S3-Net runs a real-time video scene understanding on MOMENTS. Sample visual results of S3-Net on MOMENTS are shown in Fig. 8. We observe that all objects in these frames can be located and segmented, then activities in video stream can be recognized precisely.

5.3. Performance Analysis

Besides the impressive functions and accuracy of the proposed framework, the compactness and speed are also outstanding compared to existing approaches. Table 3 shows the model size and speed comparisons among different baselines. It can be seen that S3-Net achieves an excellent compression ratio, namely, $6.9\times$ and $2.9\times$ storage reduction when compared to the original LSTM [14] and Mask R-CNN [15], respectively. The whole S3-Net costs only 89.2MB to perform both object segmentation and activity recognition with good accuracy. Moreover, S3-Net runs at 22.8 FPS on the high-resolution CityScapes, while 137.3 FPS on MOMENTS, which is considered "very fast" for both object segmentation and activity recognition tasks. Since the model size is significantly reduced and the speed is highly accelerated, the proposed S3-Net provides a turnkey solution for fast and lightweight video scene understanding, say, in autonomous driving.

Scale	Depth	AP	AP_{50}	Acc(%)	FPS
480	9	24.6	48.7	89.8	**33.1**
	12	28.2	51.5	95.1	31.3
	15	28.5	52.1	95.6	28.8
800	9	29.4	53.5	95.8	26.5
	12	32.3	57.2	98.3	22.8
	15	**32.8**	**57.9**	**98.5**	19.6

Table 4. Sub-scene Detector: Larger and deeper layers bring higher accuracy, while too large or deep layers highly slow down the speed.

Backbone	AP	AP_{50}	FPS
ResNet-101-FPN	**34.9**	**59.5**	13.4
ResNet-50-FPN	32.3	57.2	**22.8**

Table 5. Backbone Architecture: Better backbones bring expected benefits, but not all frameworks rely on the deeper networks.

COCO	AP	AP_{50}	Acc(%)
with	**32.3**	**57.2**	**98.3**
without	27.9	53.6	92.0

Table 6. Pretrained COCO Model: Pretrained model on COCO remarkably improves accuracy.

5.4. Ablation Study

We run a series of ablations to further analyze S3-Net. All experiments are valuated on CityScapes and UCF11 with the same software-hardware environments. Note that in all tables, we apply AP and AP_{50} as the object segmentation accuracy on CityScapes and Acc as the activity recognition accuracy on UCF11.

Sub-scene Detector The first concern arises from the beginning of the network. As the sub-scene detector learns important coordinates for the subsequent parts, the input frame scale and depth should be investigated. In Table 4, we compare different detectors' scales and depths. At a frame scale of 800, changing the head depth from 9 to 12 provides 2.9 AP and 2.5 Acc gains while 12 to 15 provides 0.5 AP and 0.2 Acc gains and becomes stable. Therefore, we conclude that 12 is the best choice for layer depth of the sub-scene detector. Next, setting depth to be 12, changing input frame scale from 800 to 480 provides 8.5 FPS gains, and causes 4.1 AP and 3.2 Acc losses. In practice, we apply S3-Net-800 as the default, and enable S3-Net-480 when the frame sizes are small, say, in MOMENTS.

Backbone Architecture For the backbone architecture of the single-shot segmentation model, we evaluate S3-Net with 2 different backbones: ResNet-50-FPN and ResNet-101-FPN, as shown in Table 5. The results show that replacing ResNet-101-FPN to ResNet-50-FPN provides 9.4 FPS gains, and causes 2.6 AP losses. We stress that S3-Net can get competitive accuracy with the lightweight backbone when compared with larger-scale networks. Subsequently, we employ ResNet-50-FPN as the default backbone due to its compactness.

COCO Pretrained Model Here we evaluate the impacts of the COCO pretrained model used in training. Table 6 reports the accuracy with/without COCO pretrained model. We have the observation that the COCO pretrained model provides a 4.3 AP and 6.3 Acc improvement on CityScapes and UCF11.

Inputs	Acc(%)
Raw frame data	79.7
Non-structured semantic features	92.4
Structured semantic features	**98.3**

Table 7. Structured Time-series Semantic Features: Optimized inputs of the spatio-temporal model bring expected benefits.

Structured tensorization	×	×	✓	✓
Trained quantization	×	✓	×	✓
AP	**32.6**	32.3	**32.6**	32.3
Acc(%)	76.1	75.9	**98.4**	98.3
Storage(MB)	972.5	243.1	356.8	**89.2**
FPS	2.8	3.1	22.1	**22.8**

Table 8. Tensorization and Quantization: Unique benefit of accuracy improvement under deep compression.

Structured Time-series Semantic Features The structured time-series semantic features plays an important role in the proposed spatio-temporal model for activity recognition. In Table 7, we report the Acc scores with 3 different inputs to the spatio-temporal model: **1)** raw frame data, **2)** non-structured semantic features and **3)** structured semantic features. As we can see, the proposed method gets the highest Acc among all schemes, which demonstrate its importance.

Tensorization and Quantization Finally, in Table 8, we present the ablation study on tensorization and quantization by testing different training strategies, namely, with/without quantization/tensorization. The series of evaluations demonstrate the unique benefit arises from the structured tensorization and trained quantization, namely, accuracy improvement even under deep compression.

6. Conclusion

This paper has proposed the S3-Net for fast video scene understanding. A single-shot segmentation method is proposed to quickly locate and segment the target sub-scenes, instead of handling all pixels or every object candidate in the frame. Then, an LSTM-based spatio-temporal model is built from highly structured time-series semantic features for activity recognition. Moreover, the structured tensorization and trained quantization are utilized to significantly advance the S3-Net, making it friendly for edge computing. Using the benchmarks of CityScapes, UCF11, HMDB51 and MOMENTS, S3-Net achieves a remarkable accuracy improvement of 8.1%, a storage reduction of 6.9× and an inference speed of 22.8 FPS, thereby rendering it a strong candidate for real-time video scene understanding in autonomous driving.

References

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *CVPR*, pages 859–868, 2018.

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *TPAMI*, 39(12):2481–2495, 2017.

[3] JeongYeol Baek, Ioana Veronica Chelu, et al. Scene understanding networks for autonomous driving based on around view monitoring system. In *CVPR*, pages 1074–10747, 2018.

[4] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. Yolact: Real-time instance segmentation. *arXiv preprint arXiv:1904.02689*, 2019.

[5] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, pages 6299–6308, 2017.

[6] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Multi-fiber networks for video recognition. In *ECCV*, pages 352–367, 2018.

[7] Yuan Cheng, Guangtai Huang, Peining Zhen, Bin Liu, Hai-Bao Chen, Ngai Wong, and Hao Yu. An anomaly comprehension neural network for surveillance videos on terminal devices. In *Design, Automation & Test in Europe Conference & Exhibition*, pages 1396–1401, 2020.

[8] Yuan Cheng, Guangya Li, Ngai Wong, Haibao Chen, and Hao Yu. Deepeye: A deeply tensor-compressed neural network for video comprehension on terminal devices. *ACM Transactions on Embedded Computing Systems*, 19:1–25, 2020.

[9] Marius Cordts, Mohamed Omran, and Sebastian Ramos. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, pages 3213–3223, 2016.

[10] Jifeng Dai, Kaiming He, and Sun Jian. Convolutional feature masking for joint object and stuff segmentation. In *CVPR*, 2015.

[11] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, pages 3150–3158, 2016.

[12] Naiyu Gao, Yanhu Shan, Yupei Wang, Xin Zhao, Yinan Yu, Ming Yang, and Kaiqi Huang. Ssap: Single-shot instance segmentation with affinity pyramid. In *ICCV*, pages 642–651, 2019.

[13] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, pages 447–456, 2015.

[14] Mahmudul Hasan, Amit K Roy-Chowdhury, et al. Incremental activity modeling and recognition in streaming videos. In *CVPR*, pages 796–803, 2014.

[15] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. *TPAMI*, PP(99):1–1, 2017.

[16] Martin Holder, Philipp Rosenberger, and Hermann Winner. Measurements revealing challenges in radar sensor modeling for virtual validation of autonomous driving. In *ICITS*, pages 2616–2622, 2018.

[17] Alexander Kirillov, Evgeny Levinkov, Bjoern Andres, Bogdan Savchynskyy, and Carsten Rother. Instancecut: from edges to instances with multicut. In *CVPR*, pages 5008–5017, 2017.

[18] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *ICCV*, pages 2556–2563, 2011.

[19] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. In *CVPR*, pages 2359–2367, 2017.

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755, 2014.

[21] Jingen Liu, Jiebo Luo, and Mubarak Shah. Recognizing realistic actions from videos "in the wild". In *CVPR*, pages 1996–2003, 2009.

[22] Shu Liu, Jiaya Jia, Sanja Fidler, and Raquel Urtasun. Sgn: Sequential grouping networks for instance segmentation. In *ICCV*, pages 3496–3504, 2017.

[23] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *ICCV*, pages 8759–8768, 2018.

[24] Yiding Liu, Siyu Yang, Bin Li, Wengang Zhou, Jizheng Xu, Houqiang Li, and Yan Lu. Affinity derivation and graph merge for instance segmentation. In *ECCV*, pages 686–703, 2018.

[25] Mathew Monfort, Bolei Zhou, and Sarah Adel Bargal. Moments in time dataset: one million videos for event understanding. *arXiv preprint arXiv:1801.03150*, 2018.

[26] Karen Simonyan, Andrew Zisserman, et al. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, pages 568–576, 2014.

[27] Jonas Uhrig, Marius Cordts, Uwe Franke, and Thomas Brox. Pixel-level encoding and depth layering for instance-level semantic labeling. In *GCPR*, pages 14–25, 2016.

[28] Jonas Uhrig, Eike Rehder, Björn Fröhlich, Uwe Franke, and Thomas Brox. Box2pix: Single-shot instance segmentation by assigning pixels to object boxes. In *IV*, pages 292–299, 2018.

[29] Amin Ullah, Jamil Ahmad, Khan Muhammad, Muhammad Sajjad, and Sung Wook Baik. Action recognition in video sequences using deep bi-directional lstm with cnn features. *IEEE Access*, 6:1155–1166, 2018.

[30] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, pages 20–36, 2016.

[31] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang, and Lei Li. Solo: Segmenting objects by locations. *arXiv preprint arXiv:1912.04488*, 2019.

[32] Enze Xie, Peize Sun, Xiaoge Song, Wenhai Wang, Xuebo Liu, Ding Liang, Chunhua Shen, and Ping Luo. Polarmask: Single shot instance segmentation with polar representation. *arXiv preprint arXiv:1909.13226*, 2019.

[33] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.