Video Scene Segmentation Using Tensor-Train Faster-RCNN for Multimedia IoT Systems

Cheng Dai, Xingang Liu, Laurence T. Yang, Minghao Ni, Zhenchao Ma, Qingchen Zhang and M. Jamal Deen

***Abstract*—Video surveillance techniques like scene segmentation are playing an increasingly important role in multimedia Internet-of-Things (IoT) systems. However, existing deep learning based methods face challenges in both accuracy and memory when deployed on edge computing devices with limited computing resources. To address these challenges, a tensor-train video scene segmentation scheme that compares the local background information in regional scene boundary boxes in adjacent frames, is proposed. Compared to existing methods, the proposed scheme can achieve competitive performance in both segmentation accuracy and parameter compression rate. In detail, first, an improved faster Region Convolutional Neural Network (faster-RCNN) model is proposed to recognize and generate a large number of region boxes with foreground and background to achieve boundary boxes. Then the foreground boxes with sparse objects are removed and the rest are considered as optional background boxes used to measure the similarity between two adjacent frames. Second, to accelerate the training efficiency and reduce memory size, a general and efficient training way using tensor-train decomposition to factor the input-to-hidden weight matrix, is proposed. Finally, experiments are conducted to evaluate the performance of the proposed scheme in terms of the accuracy and model compression. Our results demonstrate that the proposed model can improve the training efficiency and save the memory space for the deep computation model with good accuracy. This work opens the potential for the use of artificial intelligence methods in edge computing devices for multimedia IoT systems.**

***Index Terms*—Video scene segmentation, Multimedia IoT system, Deep learning, Tensor-train.**

I. Introduction

MULTIMEDIA Internet-of-Things (IoT) has developed as a new type of IoT, which integrates image processing, computer vision and networking capabilities. It has been widely used in surveillance, human behaviour analysis and automatic event recognition [1]. Recently, with the ever-increasing computing power and storage capacity of digital cameras, front-end surveillance equipment can now store thousands of hours of video. As far as we know, a huge amount of video data facilitate and enrich people's lives through big data analysis, but it also brings many problems in data management [2]. If the video sequences can be divided into different segments based on story scenes, then it will be more convenient for users to retrieve useful information according to their interests and hobbies [3]. Therefore, in recent years, video scene segmentation has become a hot topic in computer vision. Its goal is to divide the video stream into a set of meaningful and manage-able segments, and greatly improve people's efficiency in terms of video indexing. However, the exponential growth in the volume of publicly available digital data has created a number of challenges because of changing scenes, different characters and variable stories [4]. To overcome these challenges, more robust, efficient, and accurate video segmentation algorithms are needed.

Video scene segmentation techniques have experienced from hand-crfated feature based methods and deep learning based methods with the revolution of deep neural network [5]. However, the mechanism used in deep learning for highly non-convex optimization problem will produce large numbers of redundant parameters during the training stage. Such huge parameter training can only be implemented in high-performance workstations because they require massive computation resources. This limitation prevents the creation of efficient deep models that are fast enough for massive real-time inference or small enough to be implemented in low resource end devices [6]. To reduce the parameters, many deep model compression methods have been proposed in recent research such as network pruning and knowledge distillation. However, in these methods, training from scratch is impossible and the training process is laborious. Besides that, a recent study by Novikov et al shows that the dense weight matrices inside convolutional neural network (CNN) model can be replaced using tensor train format. With the TT-format, they significantly compress the number of parameters and kept the model accuracy degradation to a minimum [7].

In fact, there have been several tensor based compression techniques proposed from inference process. Denton et al., compressed the weight matrix of fully-connected layers using SVD with less accuracy drop in 2014 compared existing methods [8]. In recent research, more emphasis has been put on matrix quantization to make the parameter matrix more sparse, and the common used methods are hashing techniques and circulate projection. The experiments show that these methods can achieve better compression performance than previous SVD method, but they cannot speed up the convolutional inference process for incrementing data [9]. More recently, to speed up the convolutional layers, several methods based on low-rank decomposition of convolutional

Cheng Dai is with the School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, China, and he is also a visiting student in Department of Electrical Engineering and Computer Science at McMaster University, Hamilton, ON L8S4K1, Canada.

Xingang Liu and Minghao Ni are with the School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, China.

Laurence T. Yang, Zhenchao Ma and Qingchen Zhang are with the Department of Computer Science, St. Francis Xavier University, Antigonish, NS B2G2W5, Canada.

M. Jamal Deen is with the Department of Electrical Engineering and Computer Science at McMaster University, Hamilton, ON L8S4K1, Canada.

kernel tensor were proposed. The common decomposition based methods are CP decomposition, tucker decomposition and tensor-train decomposition [10]. Among them, CP decomposition cannot be successfully used in the model with deeper convolution layers because of its instability issue. However, tucker decomposition cannot get the ideal compression rate duo to the core tensors. Comparatively, tensor-train decomposition have obvious advantage in deeper convolution model compression compared with CP decomposition, and it can also achieve more desirable performance when compared with tucker decomposition. Therefore, it has attracted more and more attention in deep model compression recently.

In this paper, a tensor-train deep computation model is presented for video scene segmentation. The model is constructed based on the following aspects: on the one hand, each video frames can be extracted as moving object and background, and the objects may change in different time but the background often does not change significantly in the short sequence. On the other hand, the background information in one shot has high similarity, it can judge the change of backgrounds in a video and divide videos into several clips with the same scene. Moreover, to further meet the convergence of deep learning and multimedia IoT applications, the tensor-train mechanism is introduced in parameter learning, which maps matrix to tensor space and use a tensor factorization framework to compress parameters of the convolutional layer. The contributions of this paper can be summarized as follows: (1) To achieve desire accuracy, a new video segmentation scheme based on local background information is proposed with comparing the image information in regional scene boundary boxes. (2) To reduce the number of parameters in the conventional deep computation model, we introduce a tensor-train model to decompose the convolution operation.

The remainder of this paper is organized as follows. Section II analyzes the related achievements in the field of video scene segmentation and tensor-train applications. Section III presents the prosed segmentation method and illustrates the details of key points. In Section IV, we design experiments to verify the feasibility of our proposal and make the analysis according to the experiment results. Finally, the conclusions are given in the Section V.

II. Related Works

In this section, the preliminaries including the traditional video scene segmentation method and the tensor-train network are described.

A. Video Segmentation Algorithms

The main idea about hand-crafted based feature based method is that low level features extracted from key-frame are sent to a clustering algorithm to detect the scene boundaries [11]. The main task in this kind of methods is to explore the effective low-level feature and improve the clustering algorithms. In the early research [12], the researchers carry out the pixels difference to compute the similarity between two consecutive frames. The frame is treated as the shot boundary if the sum of pixels difference between two frame images exceeds the certain threshold [13]. After that, Ngo et al. proposed a method according to the information about motion and colour information [14]. They initially segmented the video into shots and then selected the key frames by using colour histogram, and the scene boundaries are finally detected by time constraint grouping. However, these low-level based methods have many limitations such as noises, partial occlusions and translation. Comparatively, conventional hand-crafted feature based methods are database-orientated, which means that different tasks should investigate different feature extraction algorithm [15]. Furthermore, hand-crafted extractors are of high complexity, which brings a challenge to put it into real applications.

With the development of machine learning, deep learning concept was proposed by Hinton in 2006, and has been widely applied in traditional artificial intelligence domains such as: semantic segmentation, human pose estimation, image retrieval, object detection and many more [16]. In recent years, to promote the rapid development of the related deep learning network, many related international competitions are held specifically for deep learning application, and there have been various tools and methods based on deep learning developed for different scene segmentation applications [17]. For example, S. Petscharnig applied the shot classification to medical video field. The experimental result shows that CNN-based single-frame classification approach is able to provide useful suggestions to medical experts while annotating video scenes [18]. Li et al. propose a segmentation method in traffic scenes based on RGB-D images and deep learning. It presents a new deep fully convolutional neural network architecture for scene segmentation [19]. The results show that this algorithm has a good performance . Liang et al. proposed a new approach which uses CNN model to extract features of video sequence parallelly based on GPU, so they could simplify the expression of video and reduce the calculation time for shot detection, then they took local frame similarity and dual-threshold sliding window similarity into consideration to increase recall and precision of shot detection [20].

B. Tensor-Train Decomposition

Deep learning networks have given the state-of-the-art performance on many problems in computer vision and other fields, and deeper network means better inference performance in feature learning. However, even a simple CNN model, it involves millions of parameters to learn, which brings a great challenge for transferring deep model on the pre-front multimedia IoT system with limited computation resources from workstation [21]. In the past research, a group of recent works have significant speed-ups of CNN convolutions. Among them, tensor decomposition, as one of the most efficient tools, has been widely used in computer vision and signal processing [22]. In short, tensor based methods for deep learning network compression can be classified into two categories namely tensorized inference and tensorized training [23]. For tensorized inference, it uses the idea train and then compress. The neural network is first trained with higher computation machine such as GPU cluster, and then

tensor decomposition is employed to compress the well-trained model into a smaller one to make it available to be deployed on a device with limited resource. For a simple example, Denton et al. proposed a method which compresses the weight matrix by using low-rank CP decomposition [24]. However, for tensorized training, the convolution kernel can be deemed as a 4D tensor, laying the theoretical foundation for tensor-train mechanism. Then a higher-order tensor can be represented as a set of smaller core-tensors with fewer parameters and thus realize the model size compression. Jadrberg et al. proposed a more effective decomposition method with simple tensor decomposition algorithm, which approximates 4D kernel as two 3D tensor composition, and it shows better performance in terms of model size reduction [25]. Besides that, tensorized training has been also transformed from spatial reasoning to temporal inference. In Novikov's research, they reshaped a fully connected layer into high-dimensional tensor and then factorize it using tensor-train [26]. In temporal inference, Yang et al. proposed a more general and efficient method by factorizing the input-to-hidden weight matrix using tensor-train decomposition which is trained simultaneously with the weights themselves [27].

III. Video Scene Segmentation with Deep Learning

In this section, we introduce a feasible method which will realize video segmentation based on tensor based Faster-RCNN. The whole architecture of our proposal is shown in Fig.1, where, a video frame is sent into tensor-train based Faster-RCNN model to recognize and generate a large number of region boxes with foreground and background areas. Then, the foreground boxes with objects are removed and the rests are considered as background boxes. Meanwhile, to reduce the computation, some tiny and overlapped boxes are also ignored. Finally, these reserved region boxes are applied on the next frame so that the similarity comparison rates are calculated between two neighbouring frames, and we finish video segmentation according to the result.

A. Tensor-Train Neural Network

1) Tensor-train decomposition: The tensor decomposition is scaled to many tensor with different dimensions and the calculation is demonstrated shown in Fig. 2. For example, given a $d-$dimensional target tensors $A \in R^{p_1 \times p_2 \times p_3 \times \cdots \times p_d}$, it is factorized as

$$\mathbf{A}(j_1, j_2, \cdots j_d) \overset{TT}{=} \mathbf{G}_1[j_1]\mathbf{G}_2[j_2]\mathbf{G}_3[j_3]\cdots\mathbf{G}_d[j_d]. \quad (1)$$

where $\mathbf{G}_k \in R^{p_1 \times r_{k-1} \times r_k}, j_k \in [1, p_k], \forall k \in [1, d]$ and $r_0 = r_d = 1$. The set of tensors $\mathbf{G}_{k\,k=1}^{d}$ are core-tensors.

However, if each integer p_k is factorized as $m_k.n_k$, then $\mathbf{G}_k$ is reshaped into $\mathbf{G}_k^* \in R^{m_k \times n_k \times r_{k-1} \times r_k}$ and each index can be depicted with two indices(v_k, t_k) accordingly.

$$\begin{aligned} \mathbf{G}_k(j_k) = \mathbf{G}_k^*(v_k, t_k) \in R^{r_{k-1} \times r_k} and \\ v_k = [j_k/n_k], t_k = j_k - n_k \times v_k \end{aligned} \quad (2)$$

Therefore, the tensor decomposition $\mathbf{A}$ is rewritten as follows:

$$\begin{aligned} \mathbf{A}((v_1, t_1), (v_2, t_2), \cdots (v_d, t_d)) = \mathbf{G}_1^*(v_1, t_1)\mathbf{G}_2^*(v_2, t_2) \\ \mathbf{G}_3^*(v_3, t_3)\cdots\mathbf{G}_d^*(v_d, t_d). \end{aligned} \quad (3)$$

2) Tensor-train convolutional layer: Considering that in our proposed method, a convolutional layer is transformed as matrix-by-matrix multiplication, we then reshape the matrix $\mathbf{K}$ into a tensor for construct tensor-train computation. The 4-dimensional kernel is firstly reshaped into matrix according to equation 6, and then transformed into tensor-train format according to equation 7. The 4-dimensional kernel can be transformed as follows

$$K(x, y, c', s') = G_0[x, y]G_1[c_1, s_1]\cdots G_d[c_d, s_d] \quad (4)$$

where, considering that C is factorized into $\prod_{i=1}^{d} C_i$, S is factorized into $\prod_{i=1}^{d} S_i$, and c' and s' are interfered as $c' = c_1 + \sum_{i=2}^{d}(c_i - 1)\prod_{j=1}^{i-1} C_i$, $s' = s_1 + \sum_{i=2}^{d}(s_i - 1)\prod_{j=1}^{i-1} S_j$. The tensor-train layer is transformed accordingly as follows:

$$\begin{aligned} \tilde{y}(x, y, s_1, s_2 \cdots, s_d) = \sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\sum_{c_1 \cdots c_d} \tilde{\mathcal{X}}(x + i - 1, y + j - 1, c) \\ G_0[x, y]G_1[c_1, s_1]\cdots G_d[c_d, s_d] \end{aligned} \quad (5)$$

3) Tensor-train Network: A linear transformation of the input vector x in convolution is:

$$y = Wx + b \quad (6)$$

It can be expressed in scalar form as:

$$\begin{aligned} y(j) = \sum_{i=1}^{M} W(i, j)x(i) + b(j) \\ \forall j \in [1, N], x \in \mathbb{R}^M, y \in \mathbb{R}^N \end{aligned} \quad (7)$$

Then M and N can be factorized into two arrays with the same length, which are $M = \prod_{k=1}^{d} m_k$ and $N = \prod_{k=1}^{d} n_k$. After that, the input x and output y can be implemented TT-decomposition and reshaped into $\mathcal{X} \in \mathbb{R}^{m_1 \times m_2 \times \ldots \times m_d}$, $\mathcal{Y} \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_d}$. Finally, the linear transformation can be denoted in tensor format as:

$$\begin{aligned} \mathcal{Y}(j_1, j_2, ..., j_d) = \sum_{i_1=1}^{m_1}\sum_{i_2=1}^{m_2}\ldots\sum_{i_d=1}^{m_d} \mathcal{W}((i_1, j_1), (i_2, j_2), ...(i_d, j_d))\times \\ \mathcal{X}(i_1, i_2, ..., i_d) + \mathcal{B}(j_1, j_2, ..., j_d) \end{aligned} \quad (8)$$

Here $\mathcal{W}$ is represented through TT-format with cores $G_k[i_k, j_k]$, and the transformation can be expressed in tensor form:

$$\mathcal{Y}(i_1, ..., i_d) = \sum_{j_1, \ldots, j_d} G_1[i_1, j_1]...G_d[i_d, j_d]\mathcal{X}(j_1, ..., i_d) + \mathcal{B}(i_1, ...i_d) \quad (9)$$

Meanwhile, to update the parameters in tensor $\mathcal{X}$, $\mathcal{B}$ and matrices $G_k[i_k, j_k]$, we need to compute partial derivative respectively and achieve gradients. The $\frac{\partial L}{\partial x}$, $\frac{\partial L}{\partial b}$ and $\frac{\partial L}{\partial G_k[i_k, j_k]}$ are calculated by the following equations:

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial x} = W^T\frac{\partial L}{\partial y} \quad (10)$$

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial b} = \frac{\partial L}{\partial y} \quad (11)$$

$$\frac{\partial L}{\partial G_k[i_k, j_k]} = \sum_{i \setminus k}\frac{\partial L}{\partial y(i)}\frac{\partial y(i)}{\partial G_k[i_k, j_k]} \quad (12)$$

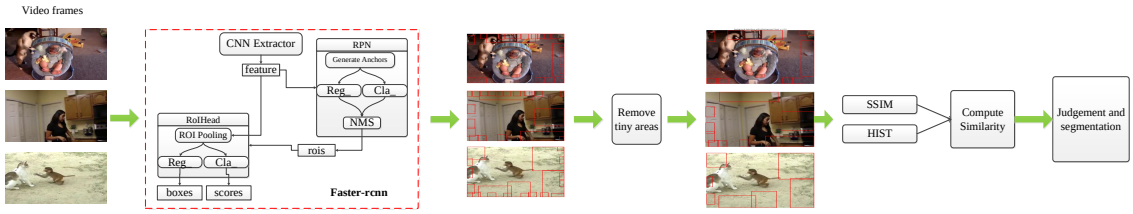

Fig. 1. The whole architecture of the proposed faster-RCNN

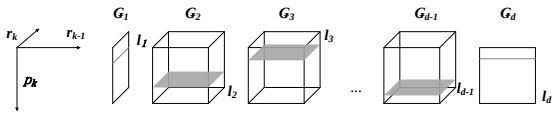

Fig. 2. Tensor-Train Decomposition Model: One tensor can be performed as a sequence of vector-matrix-vector multiplication to reconstruct the target tensor

Because $y(i)$ is expressed in the form:

$$y(i) = \sum_{j} G_1[i_1, j_1]...G_k[i_k, j_k]...G_d[i_d, j_d]x(j) + b(i) \quad (13)$$

after derivative, the entry of $G_k[i_k, j_k]$ will be eliminated and the final results of $\frac{\partial L}{\partial G_k[i_k, j_k]}$ is:

$$\begin{aligned} \frac{\partial L}{\partial G_k[i_k, j_k]} &= \sum_{j \setminus k} G_1[i_1, j_1]...G_{k-1}[i_{k-1}, j_{k-1}] \\ &\quad \times G_{k+1}[i_{k+1}, j_{k+1}]...G_d[i_d, j_d]x(j) \\ &= \sum_{j \setminus k,d} G_1[i_1, j_1]...G_{k-1}[i_{k-1}, j_{k-1}] \\ &\quad \times G_{k+1}[i_{k+1}, j_{k+1}]...G_{d-1}[i_{d-1}, j_{d-1}] \\ &\quad \times \sum_{j_d} G_d[i_d, j_d]x(j) \end{aligned} \quad (14)$$

In the training process, we treat tensor-train core elements as the parameters of tensor-train layer and use stochastic gradient descent with momentum to them.

B. Background Regions Generation Based on Faster-RCNN

Accordingly, a picture can generally be segmented into two parts namely foreground and background. However, in experiments, the generated background region boxes have different sizes and some region boxes may overlap in the actual operation. According to the previous research, the problem mentioned above will brings two challenges in scene segmentation. The first one is that small region boxes contain little information, and they are unable to represent the key part in the frames. Another problem is that too many overlapped areas will cause the similarity of certain area computed repeatedly. Therefore, we plan to remove those tiny and overlapped regions to ensure similarity comparison performance and reduce computation complexity. In this process, we use areas and Intersection over Union (IoU) as the criteria for evaluating the algorithm. If the area of the background region box is less the certain threshold, the region box will be considered as the tiny regions and discarded. Similarly, if two region boxes are overlapped and the IoU exceeds the certain threshold, the box with smaller area will be removed. The area and IoU of the region boxes are defined as:

$$A_i = (x_2^i - x_1^i) \times (y_2^i - y_1^i) \quad (15)$$

$$IoU = \frac{Area|(B_{box_i}) \bigcap (B_{box_j})|}{Min(Area|B_{box_i}|, Area|B_{box_j}|)} \quad (16)$$

where x_1^i and x_2^i are the left and right x-axis coordinate of the i^{th} boxes, y_1^i and y_2^i are the upper and lower y-axis coordinates of the i^{th} boxes, A_i is the area of the i^{th} box, B_{box_i} and B_{box_j} are i_{th} and j_{th} boxes respectively. The procedure of this algorithm, shown in the Fig.1, is depicted as following

Algorithm 1 Remove Tiny and Overlapped Boundary Boxes

Input: set of background boundary boxes $B_{box_{i=1}^m}$, each $B_{box_{i=1}^m=(x_1^i, y_1^i, x_2^i, y_2^i)}$

Output: Larger area boxes

```
for i = 1;i < m;i++ do
    Area=(x_2^i − x_1^i) * (y_2^i − y_1^i)
    if Area < threshold_1 then
        remove B_{box_{i=1}}
    end if
end for
Obtain a new set of boundary boxes B_{box_{i=1}^k}
for each i ∈ [1, k],j ∈ [1, k],i ≠ j do
    IoU= Area|(B_{box_i}) ⋂ (B_{box_j})| / Min(Area|B_{box_i}|,Area|B_{box_j}|)
    if IoU > threshold_2 then
        Remove Min(B_{box_i}, B_{box_j})
    end if
end for
Obtain the final larger area boxes
```

C. Similarity Comparison Module

After preprocessing of background and foreground, the features are sent to similarity comparison module to calculate the similarity between two adjacent frames. The similarity comparison module mainly consists of two parts namely structural similarity module and histogram similarity module. In the next, we will describe it in detail.

1) Structural similarity module: Structural similarity is an index used to measure the similarity between two images. It introduces an alternative complementary framework for quality assessment based on the degradation of structural information. In general, the structural information refers to luminance and contrast, and the whole assessment system consists of three comparison modules: luminance comparison, contrast comparison and structure comparison in our prosed TT faster RCNN. The main architecture of structural similarity is presented in Fig. 3 below.

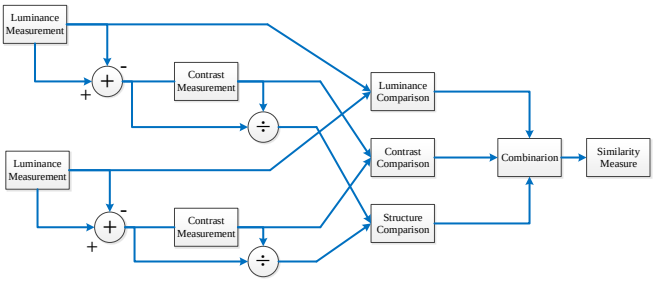

Fig. 3. The algorithm flowchart of structural similarity

here structure similarity ssim(x,y) will satisfy the following equation which combines three comparison functions namely luminance, contrast and structure respectively.

$$ssim(x,y) = L(x,y) \times C(x,y) \times S(x,y) \tag{17}$$

where L(x,y) is luminance comparison, C(x,y) is contrast comparison and S(x,y) is structure comparison, and the result is computed as follows $L(x,y) = \frac{2\mu_x\mu_y+C1}{\mu_x^2+\mu_y^2+C1}$, $C(x,y) = \frac{2\sigma_x\sigma_y+C2}{\sigma_x^2+\sigma_y^2+C2}$,$S(x,y) = \frac{\sigma_{xy}+C3}{\sigma_x\sigma_y+C3}$. Where $C1$, $C2$ and $C3$ are constants to avoid that the denominator is 0, and $C1 = (K1 * L)^2$, $C2 = (K2 * L)^2$, $C3 = C2/2$. In general, $K1 = 0.01$, $K2 = 0.03$, $L = 255$ (the dynamic range of pixel values, and it is usually defined as 255). Meanwhile, the parameter μ_x, σ_x, σ_{xy} are computed as following equations.

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{18}$$

$$\sigma_x = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \sigma_x)^2} \tag{19}$$

$$\sigma_{xy} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)(y_i - \mu_y) \tag{20}$$

where N is the number of the pixels of an image and x_i is the pixel value μ_x is the mean intensity, σ_x is the standard deviation and σ_{xy} is the covariance of two images.

Originally, the structure similarity index is used to assess the image quality for two images with same contents. For neighbouring frames in videos, the contents of them are almost the same because whether the foreground or background will not change in such a short interval, which means that the structure similarity index may get high values between two continuous frames with the same scene.

2) Histogram similarity module: Histogram similarity is another common method used in image comparison. For two images x and y, we first calculate their respective histograms. Then, the Euclidean metric is used to compute the normalized correlation coefficient of the two histograms,

$$hist(x,y) = \frac{1}{N}\sum_{i=1}^{N}(1 - \frac{|x_i^h - y_i^h|}{Max(x_i^h, y_i^h)}) \tag{21}$$

where x_h and y_h are the histograms of two images. In general, histogram can be well normalized into 256 grayscale values and it only reflects the probability distribution of the image pixel values. However, this kind of index represents similarity on colours rather than contents. It may play a role in judging the background but it will also cause similarity results inaccuracy in some cases. A good solution for it is based on computing structure similarity and the histogram similarity is used as supplementary. It will give consideration to both content and colour information, which makes the similarity comparison more comprehensive. The details about this similarity computation solution will be given in next part.

D. Similarity Algorithm Design

We use both the SSIM and histogram similarity as the index and intend to compromise them as a new standard to judge changes of frame scenes. For arbitrary two neighbouring frames, the background region boxes will be generated on the first frame and applied on the both two frames, so they have the same boundary boxes. Then we perform similarity calculation of two frames on contents of each box to obtain the s_i and h_i. Because background region boxes have different areas and the area may determine the amount of information in this box, we decide to give weights to each region box according to their areas

$$w_i = \frac{A_i}{\sum_{i=1}^{N} A_i} \tag{22}$$

where A_i is the area of the i^{th} box. With these weights, the total ssim and histogram similarity between two neighbouring frames can be computed using

$$S = \sum_{i=1}^{n} s_i \times w_i; H = \sum_{i=1}^{n} h_i \times w_i \tag{23}$$

where s_i and h_i are the ssim and histogram similarity value of the i^{th} region box respectively. To combine them into a new similarity standard, we compute the harmonic mean of S and H as the final similarity

$$Similarity = \frac{2 \times S \times H}{S + H} \tag{24}$$

The harmonic mean may be affected easily by the extreme value, especially extreme minimum value. We are more focused on the smaller one between S and H, because the lower the similarity, the more likely the background of frames will change. If one of S and H is smaller than the threshold, that means the high possibility of background change. Both S and H are the similarity indices so the lower one should have more influence factors.

IV. Experiment Verification and Analysis

A. Experiment Setting

VOC datasets are representative datasets which are widely used to evaluate the performance of faster-RCNN deep model. To train our model, we use VOC2007 and VOC2012 as experimental data to evaluate our tensor-train deep model in terms of detection accuracy and model compression rate [28]. In our experiment, we randomly select 60% of the data as the training set while the remaining 40% dataset are used as testing dataset. Moreover, to verify its robustness and segmentation performance of the presented scheme, we randomly select four different videos from different sources such as MSR-VTT datasets, MSVD datasets and Youtube as the test data. The proposed joint network is trained on a single GTX Tian 1080 with 32 GB memory with python. The initial learning rate of joint network is set as 0.001, and the joint network weight parameters are trained and adjusted by the mini-batch SGD with the momentum parameter set as 0.9. For training, we initialize the parameters of tensor-train layer with random noise.

B. Parameter Threshold Design for Similarity Comparison

In the experiment setting process, there are two problems to be considered. The first problem is how to determine the area threshold when we remove tiny boundary boxes. The generated boundary boxes are very disordered and have various sizes especially for the video sequences with complex scene change. To set a suitable threshold value, 4 groups of experiments are conducted on 4 different images. The number of the rest boxes are compared with an area threshold range from 100 to 1000. The flow chart shows that comparison results in Fig. 4. We find that the number of four different images is decreasing with the area of the box increase, and the number keep stable after the area increase to 800.

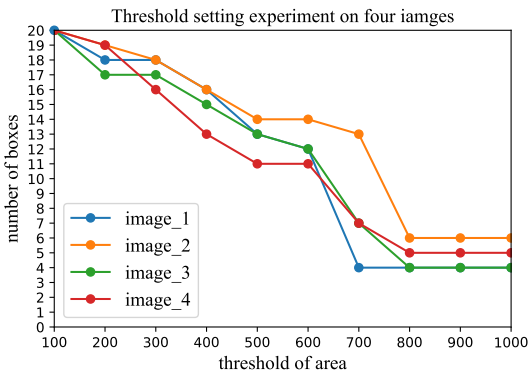

Fig. 4. The performance comparison with different area thresholds

The second problem is to design a suitable threshold of the similarity between local regions of two neighbouring frames. In our experiment, the first two groups are the comparison of neighbouring frames with the same scene, but the third frame group represents shot boundaries. For continuous scene content, the similarity is always much higher than that of changing scenes. However, the similarity of sports video frame in a shot is little lower because of its fast movement. According to the table I, we can find that similarity value is over 50% and the similarity value is lower than 40%. Therefore, we select the average value and set the threshold of similarity as 0.45 according to the results.

TABLE I
EXAMPLES OF SIMILARITY BETWEEN SCENES

	Sports	News	Daily	Movie
1(same scene)	0.584	0.820	0.784	0.911
2(same scene)	0.600	0.767	0.739	0.877
3(different scene)	0.172	0.400	0.286	0.273

C. Results of Our Model with Tensor-Train Network

In VGG-16, the number of parameters of last six convolution layer with 512 output channels occupy over 80% parameters in convolution computing. Meanwhile, parameters of fully-connected layers occupy approximate 80% parameter of CNN, and fully-connected layers of CNN is the memory bottleneck in the neural network parameter compression research. To compress the network, we replace these layers with TT-layer in our experiment. Considering that the rank of a tensor will directly affects the size of well-trained deep learning model, we test different ranks with the VOC 2007 and VOC2012. We train the network in two stages. First, we train the convolutional layer with normal convolution and record the model size. Then we replace the last six layers and three fully connected layers with TT-conv, and train it with different ranks. To find the suitable rank for further feature extraction, we compare the proposed tensor-train convolution with the original convolution with the same model corresponding to different choices of TT-ranks.

TABLE II
EXAMPLES OF SIMILARITY BETWEEN SCENES

dataset	VOC07	VOC12	VOC(07+12)		
	Acc.	Acc.	Acc.	para.	compr.
conv(baseline)	70.8%	69.3%	75.7%	138365992	1
TT-conv(32)	70.1%	68.9%	74.9%	8056256	17.2
TT-conv(16)	68.8%	65.3%	70.5%	3320960	41.7
TT-conv(8)	43.3%	37.7%	45.8%	2136800	64.8
TT-conv(4)	24.6%	19.3%	28.7%	1840592	75.2
TT-conv(2)	13.4%	12.6%	15.5%	1766504	78.3

Table II shows the accuracy and compression rate with the rank value from 2 to 32. When the value of rank is set as 32, our tensor network can approximate the original deep model in terms of accuracy and the memory size. However, such an accurate approximate tensor is not required in the convergence of artificial intelligence and multimedia IoT. With the rank value of 16, the detection accuracy losses are limited to 5%, and the compression rate of deep learning model increase average $41.7\times$. While when we use a tensor with rank 8, the accuracy drops significantly. Considering that the detection accuracy will directly affect the final segmentation performance, we select the rank of 16 as the baseline tensor-train model for further segmentation experiment.

For the convolutional layers, the number of parameters equals to l^2CS, where l is the size of the kernel, C is the

input channel and S is the output channel. Therefore, the number of parameters in the last 6 convolutional layers is 12,985,344 and it occupies 88.2% of the total parameters in all convolutional layers (14,723,136). The original parameters in fully-connected layer can be computed by $M * N$, where M is the input channel and N is the output channel. The numbers of three fully-connected layer are $25088 * 4096 + 4096 = 102,764,544$, $4096 * 4096 + 4096 = 16,781,312$ and $4096 * 1000 + 1000 = 4,097,000$, which occupy over 85% in total networks. To implement TT decomposition in fully-connected layers, we reshape the 25088-dimensional input vectors to the tensors of the size $2 \times 7 \times 8 \times 8 \times 7 \times 4$ and 4096-dimensional output vectors to the tensors of the size $4 \times 4 \times 4 \times 4 \times 4 \times 4$. The third fully-connected layer output is 1000-dimensional which is reshaped to $2 \times 5 \times 2 \times 5 \times 2 \times 5$. And we set TT-ranks here are $r = \{r_0 = 1, r_1 = 16, r_2 = 16, r_3 = 16, r_4 = 16, r_5 = 16, r_6 = 1\}$, the number of parameters after TT decomposition is expressed as:

$$N = \sum_{k=1}^{d} m_k n_k r_{k-1} r_k \tag{25}$$

So the number of parameters in first FC layer is 31,104, the number in second FC layer is 16,896 and in third is 14784. The compression of these layers reaches $3303\times$, $993\times$ and $277\times$.

The TT-decomposition reduce the parameters in convolutional layers from l^2CS to $O(R^2(2l + C + S))$. Therefore, the parameters in last 6 convolutional layers are $198144 + 5 * 253680 + 1024 * 6 = 1,522,688$, which is compressed by $8.5\times$.

After we implement TT decomposition in the last 6 convolutional layers and three fully-connected layers, the total parameters in network are 3,314,816 and is compressed by $138365992/3320960 = 41.7\times$.

D. Results of Video Segmentation with Our Model

The similarity standard we designed shows a large difference between the same and different scenes. It can be used to judge the shot boundaries. The results of our experiment contain original shots, correct shots, missed shots and false shots. Original shots are divided artificially which are ground truth while correct, missed and false shots are statistical data generated by the algorithm. Here we use recall and precision these two indexes to express accuracy. The recall is ratio of correct shots and origin shots and the precision is the ratio of correct shots and all the shots segmented.

$$\text{recall} = \frac{\text{number of correct shots}}{\text{number of origin shots}} \tag{26}$$

$$\text{precision} = \frac{\text{number of correct shots}}{\text{number of correct shots + number of false shots}} \tag{27}$$

F-score is also to compare accuracy between different boundary box selection methods.

$$F - \text{score} = \frac{\text{recall} \times \text{precision}}{\text{recall} + \text{precision}} \times 2 \tag{28}$$

The segmentation results of 4 video classes are displayed in table III, containing segmentation recall and precision with three different deep models. In the fixed boundary, 12 background region boxes are selected on the edge of frames and distributed uniformly. From the Table, we can find that the selected boxes generated by our proposal with normal convolution can achieve the best performance for four different video test sequence from simple to complex. For most videos, their recall and precision index can keep around 90 percent sometimes even up to 100 percent using faster-RCNN. There are few mistakes during the detection and segmentation process. However, for sports videos, their scenes are very complex because a short video contains many scene changes. The 5-frame interval may also affect the similarity results because it makes the scene change more quickly. Meanwhile, our proposal with tensor-train model has worse performance compared with the normal convolution but better with the fixed boxes. Because, for most videos, the main characters gather around the centre of the images and the scenes are distributed all around, especially, in the left, right and upper edges. The fixed boundary boxes seem more conservative. Though it ensures that most contents in boxes are background parts, sometimes it misses much background information in other locations. Fig. 5 presents some visualized results of scene segmentation

TABLE III
RESULTS COMPARISON WITH FIXED BOUNDARY BOX

dataset	fixed boundary		conv-baseline		Tensor-train model	
Movie	Recall	Precision	Recall	Precision	Recall	Precision
1	82.1%	67.0%	100%	87.5%	91.5%	91.3%
2	82.9%	73.9%	97.6%	90.9%	87.2%	82.8%
3	79.3%	74.2%	96.6%	93.3%	87.8%	85.3%
4	74.0%	67.3%	94.0%	90.4%	88.1%	82.9%
Sports	Recall	Precision	Recall	Precision	Recall	Precision
1	70.0%	55.3%	90.0%	73.0%	89.6%	72.1%
2	71.4%	52.6%	89.3%	73.5%	84.8%	71.8%
3	78.6%	57.9%	85.7%	72.7%	83.1%	73.2%
4	74.2%	57.5%	83.9%	78.8%	86.1%	77.6%
News	Recall	Precision	Recall	Precision	Recall	Precision
1	84.6%	73.3%	92.3%	92.3%	83.5%	77.4%
2	75.0%	68.1%	85.0%	100.0%	87.9%	97.6%
3	76.0%	73.1%	100.0%	93.5%	83.4%	92.3%
4	71.4%	71.4%	81.0%	100.0%	83.5%	88.1%
Daily life	Recall	Precision	Recall	Precision	Recall	Precision
1	100.0%	71.4%	100.0%	100.0%	100.0%	100.0%
2	85.7%	66.7%	100.0%	100.0%	98.3%	100.0%
3	87.5%	77.8%	100.0%	100.0%	97.8%	96.7%
4	85.7%	75.0%	85.7%	100.0%	89.1%	98.2%

After computing the F-score, the comparison of the accuracies of three related methods is listed in Table IV. The accuracy of video shot segmentation with the fixed boundary box is not as good as our proposal with normal convolution. With the utilization of our dynamic box selection method, accuracy gets a significant improvement. The accuracy of video shot segmentation with faster-RCNN is always higher than that of fixed boundary box and it overall improves about 15-20 percent. It means that generating background boxes with deep learning is more reasonable than fixed boxes. For our proposal with TT-conv, the F-score is lower than with normal convolution. Because a large number of parameters are discarded during the tensor decomposition process. However,

Fig. 5. Video sample images of our proposed model with different databases in the experiment

TABLE IV
F-SCORE COMPARISON BETWEEN THREE METHODS

F-score	Movie			
videos	1	2	3	4
Faster-Rcnn	93.3%	94.1%	94.9%	92.2%
Our Proposal	81.6%	87.3%	90.1%	90.3%
Fixed	73.8%	78.1%	76.7%	70.5%
F-score	Sports			
videos	1	2	3	4
Faster-Rcnn	80.6%	80.6%	78.7%	81.3%
Our Proposal	71.2%	73.6%	68.7%	67.4%
Fixed	61.8%	60.6%	66.7%	64.8%
F-score	News			
videos	1	2	3	44
Faster-Rcnn	92.3%	91.9%	96.6%	89.5%
Our Proposal	91.3%	81.8%	83.7%	77.4%
Fixed	78.5%	71.4%	74.5%	71.4%
F-score	Daily life			
videos	1	2	3	4
Faster-Rcnn	100.0%	100.0%	100.0%	92.3%
Our Proposal	91.3%	97.6%	90.2%	90.0%
Fixed	83.3%	75.0%	82.4%	80.0%

compared with the fixed boxed method, our proposal with tensor-train mechanism still has better performance, which means our algorithm can improve video shot segmentation results greatly in terms of accuracy. Besides that, considering the sparsity of parameters, our tensor-train model has competitive performance if deployed in the pre-front device with limited resource.

V. CONCLUSION

In this paper, we have presented a video segmentation approach based on local background similarity, which segments videos into several shots according to different scenes with desire accuracy and less computation resource consuming. The leading property of the proposed model is that we fully utilized the powerful learning ability of deep learning and combined it with the conventional scene segmentation algorithm. This proposed model has obvious advantages in segmentation performance compared with the latest video segmentation algorithms. Furthermore, a general and efficient training way is proposed which uses tensor-train decomposition to factorize the input-to-hidden weight matrix to achieve parameter compression. Experiments have demonstrated that our approach can achieve better performance in terms of segmentation accuracy and memory efficiency. However, the interference efficiency in real application has not been considered in our research. We will focus on the efficiency optimization to make edge side have computation ability in further research, and thus has the potential to realize the convergence of deep learning based video segmentation and multimedia IoT applications.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China under grant 61872404 and the Aplied Basic Research Key Programs of Science and Technology Department of Sichuan Province on the grant 2018JY0023.

REFERENCES

[1] E. Ahmed, I.Yaqoo et al., "Internet-of-things-based smart environments: state of the art, taxonomy, and open research challenges," *IEEE Wireless Commun.*, vol. 23, no. 5, pp. 10-16, Oct. 2016.
[2] M. Silva, D. Cerdeira, S. Pinto, and T. Gomes, "Operating Systems for Internet of Things Low-End Devices: Analysis and Benchmarking," *IEEE Internet of Things J.* vol. 6, no. 6, pp. 10375-10383, Sep. 2019.
[3] A. G. Garcia, S. O. Escolano, S. Oprea, V. V. Martinez, P. M. Gonzalez and J. G. Rodriguez, "A survey on deep learning techniques for image and video semantic segmentation," *Appl. Soft. Comput.*, vol. 70, pp. 41-65, Sep. 2018.
[4] T. Le, and A. Sugimoto, "Video Salient Object Detection Using Spatiotemporal Deep Features," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5002-5015, Jun. 2018.

[5] X. Ye, J. Yang, X. Sun, Kun Li, C. Hou, and Y. Wang, "Foreground-Background Separation From Video Clips via Motion-Assisted Matrix Restoration," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 11, pp. 1721-1734, Jan. 2015.

[6] Y. Cheng, D. Wang, P. Zhou, T. Zhang, "A Survey of Model Compression and Acceleration for Deep Neural Networks," *IEEE Signal Process. Mag.*, arXiv:1710.09282.

[7] Q. Zhang, L. T. Yang, Z. Chen and P. Li, "A Tensor-Train Deep Computation Model for Industry Informatics Big Data Feature Learning," *IEEE Trans. Ind. Inform.*, vol. 14, No. 7, pp. 3197-3204, July. 2018.

[8] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," *Adv. Neural. Inf. Process. Syst.*, vlo. 27, pp. 1269-1277, Nov. 2014.

[9] Y. W. Q. H. Jiaxiang Wu, C. Leng, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, pp. 4820-4828, Jun. 2016.

[10] T. Choudhary, V. Mishra, A. Goswami and J. Sarangapani, "A comprehensive survey on model compression and acceleration," *Artif. Intell. Rev.*, Feb. 2020. DoI:10.1007/s10462-020-09816-7.

[11] L. Long, Y. Cao, T. Jiang, and Q. Zhang, "Edge Computing Framework for Cooperative Video Processing in Multimedia IoT Systems," *IEEE Trans. Multimedia*, vol. 20, no. 5, pp. 1126-1139, May. 2018.

[12] F. J. D. Pernas, M. M. Zarzuela, M. A. Rodriguez and D. G. Ortega, "Double recurrent interaction based neural architecture for color natural scene boundary detection and surface perception," *Appl. Soft Comput.*, vol. 21, pp. 250-264, Aug. 2014.

[13] A. E. Hassanien, H. A. Qaheri, and E. S. A. E. Dahshan, "Prostate boundary detection in ultrasound images using biologically-inspired spiking neural network," *Appl. Soft Comput.*, vol. 11, pp. 2035-2041, Mar. 2011.

[14] C. W. Ngo, T. C. Pong, R. T. Chin and H. J. Zhang, "Motion Characterization by Temporal Slices Analysis," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recog.*, vol. 2, pp. 768-773, Jun. 2000.

[15] M. Delakis, G. Gravier and P. Gros, "Audiovisual integration with Segment Models for tennis video parsing," *Comput. Vis. Image Und.*, vol. 111, pp. 142-154, Aug. 2008.

[16] C. Dai, X. Liu, J. Lai, P. Li, and H. C. Chao, "Human Behavior Deep Recognition Architecture for Smart City Applications in the 5G Environment," *IEEE Netw.*, vol. 33, No. 5, pp. 206-211, Sep. 2019.

[17] H.Jiang, G. Zhang, H. Wang and H. Bao, "Spatio-Temporal Video Segmentation of Static Scenes and Its Applications," *IEEE Trans. Image Process.*, vol. 17, no. 1, pp. 3-15, Jan. 2015.

[18] S. Petscharnig and K. Schoffmann, "Deep Learning for Shot Classification in Gynecologic Surgery Videos," in *Int. Conf. Multimedia Modeling*, pp. 702-713, Dec. 2016.

[19] L. Li, B. Qian, J. Lian, et al, "Traffic Scene Segmentation Based on RGB-D Image and Deep Learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, pp. 1664-1669, May. 2018.

[20] R. Liang, Q. Zhu, H. Wei, et al, "A Video Shot Boundary Detection Approach Based on CNN Feature ," in *Proc. Int. Symposium Multimedia*, pp. 168-186, Dec. 2017.

[21] Z. Huang, X. Xu, J. Ni, H. Zhu, and C. Wang, "Multimodal Representation Learning for Recommendation in Internet of Things," *IEEE Internet of Things J.* vol. 6, no. 6, pp. 10675-10685, Dec. 2019.

[22] Z. Ma, L. T. Yang, and Q. Zhang, "Support Multi-Mode Tensor Machine for Multiple Classification on Industrial Big Data", *IEEE Trans. Ind. Inform.*, DOI: 10.1109/TII.2020.2999622. Jun. 2020.

[23] A. Singh, G. S. Aujla, S. Garg, G. Kaddoum and G. Singh, "Deep Learning-based SDN Model for Internet of Things: An Incremental Tensor Train Approach," *IEEE Internet of Things J.* DOI: 10.1109/JIOT.2019.2953537.

[24] E. L. Denton, W. Zaremba, J. Bruna, Y. Le.Cun, and R. Fergus. "Exploiting linear structure within convolutional networks for efficient evaluation,"in *Proc. Advances in Neural Inform. Process. Syst.* pp. 1269-1277, 2014.

[25] M. Jaderberg, A. Vedaldi and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions." arXiv:1405.3866, 2014.

[26] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov. "Tensorizing neural networks," in *Advances in Neural Inform. Process. Syst.* pp. 442-450, 2015.

[27] Y. Yang, D. Krompass, and V. Tresp, "Tensor-Train Recurrent Neural Networks for Video Classification," in *Int. Conf. Mach. Learn.* pp. 3891-3900, 2017.

[28] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *Int. Conf. Comput. Vis. Pattern Recog.* pp. 779-788, 2016.

Cheng Dai is currently a PhD candidate in the school of Information and Communication Engineering of University of Electronic Science and Technology of China (UESTC), Chengdu, China. He is now a visiting student in the Department of Electrical and computer Engineering of McMaster University. His research interests include human action analysis, deep learning and so on.

Xingang Liu [M10-SM17] is currently a professor with the School of Information and Communication Engineering, University of Electronic Science and Technology of China (UESTC). His research interests include video transcoding, quality measurement, human identification, face recognition, 2-D/3-D video codec, and so on. Dr. Liu is a Senior Member of IEEE, and a Fellow of IET. He won the IEEE TCSC Award for Excellence for Early Career Researchers in 2016 because of his achievement on scalable coding for multimedia signals.

Laurence T. Yang [M'97, SM'15, F'20] received the B.E. degree in computer science and technology and B.Sc. degree in applied physics both from Tsinghua University, Beijing, China, and the Ph.D. degree in computer science from the University of Victoria, Victoria, BC, Canada. He is a Professor with the Department of Computer Science, St. Francis Xavier University, Antigonish, NS, Canada. His research has been supported by the National Sciences and Engineering Research Council, Canada, and the Canada Foundation for Innovation. His research interests include Cyber-Physical-Social Systems, Parallel and Distributed Computing, Embedded and Ubiquitous Computing, and Big Data.

Minghao Ni is currently a master student in the school of Information and Communication Engineering of University of Electronic Science and Technology of China (UESTC), Chengdu, China. His research interests include human behavior recognition, deep learning model compression, machine learning and so on.

Zhenchao Ma received the B.E. degree in computer science from Huazhong University of Science and Technology, Wuhan, China. He is currently working towards the M.S. degree in the Department of Computer Science, St. Francis Xavier University, Antigonish, NS, Canada. His research interests include Big Data Mining and Cyber-Physical-Social Systems.

Qingchen Zhang received the B.Eng. degree in Southwest University, China, and the Ph.D. degree in Dalian University of Technology, China. He is currently an assistant professor in St. Francis Xavier University. His research interests include Aritificial Intelligence and Smart Medicine.

M. Jamal Deen [Fellow'02] is a Distinguished University Professor and Senior Canada Research Chair in Information Technology at McMaster University, Hamilton, Canada. He is currently the Past President of the Academy of Science, Royal Society of Canada. His research interests include nano-/opto-electronics, nanotechnology, data analytics, and their emerging applications in health and environmental sciences.