CSVideoNet: A Real-time End-to-end Learning Framework for High-frame-rate Video Compressive Sensing

Kai XU Fengbo Ren
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University, Tempe AZ 85281
{kaixu, renfengbo}@asu.edu

Abstract

This paper addresses the real-time encoding-decoding problem for high-frame-rate video compressive sensing (CS). Unlike prior works that perform reconstruction using iterative optimization-based approaches, we propose a non-iterative model, named "CSVideoNet", which directly learns the inverse mapping of CS and reconstructs the original input in a single forward propagation. To overcome the limitations of existing CS cameras, we propose a multi-rate CNN and a synthesizing RNN to improve the trade-off between compression ratio (CR) and spatial-temporal resolution of the reconstructed videos. The experiment results demonstrate that CSVideoNet significantly outperforms state-of-the-art approaches. Without any pre/post-processing, we achieve a 25dB Peak signal-to-noise ratio (PSNR) recovery quality at 100x CR, with a frame rate of 125 fps on a Titan X GPU. Due to the feedforward and high-data-concurrency natures of CSVideoNet, it can take advantage of GPU acceleration to achieve three orders of magnitude speed-up over conventional iterative-based approaches. We share the source code at https://github.com/PSCLab-ASU/CSVideoNet.

1. Introduction

High-frame-rate cameras are capable of capturing videos at frame rates over 100 frames per second (fps). These devices were originally developed for research purposes, e.g., to characterize events that occur at a rate that traditional cameras are incapable of recording in physical and biological science. Some high-frame-rate cameras, such as Photron SA1, SA3, are capable of recording high resolution still images of ephemeral events such as a supersonic flying bullet or an exploding balloon with negligible motion blur and image distortion artifacts. However, due to the complex sensor hardware designed for high sampling frequency, these types of equipment are extremely expensive (over tens of thousand dollars for one camera). The high cost limits the field of their applications. Furthermore, the high transmission bandwidth and the large storage space associated with the high frame rate challenges the manufacture of affordable consumer devices. For example, true high-definition-resolution (1080p) video cameras at a frame rate of 10k fps can generate about 500 GB data per second, which imposes significant challenges on existing transmission and storage techniques. Also, the high throughput raises energy efficiency a big concern. For example, "GoPro 5" can capture videos at 120 fps with 1080p resolution. However, the short battery life (1-2 hours) has significantly narrowed their practical applications.

Traditional video encoder, e.g., H.264/MPEG-4, is composed of motion estimation, frequency transform, quantization, and entropy coding modules. From both speed and cost perspectives, the complicated structure makes these video encoder unsuitable for high-frame-rate video cameras. Alternatively, compressive sensing (CS) is a much more hardware-friendly acquisition technique that allows video capture with a sub-Nyquist sampling rate. The advent of CS has led to the emergence of new image devices, e.g., single-pixel cameras [6]. CS has also been applied in many practical applications, e.g., accelerating magnetic resonance imaging (MRI) [13]. While traditional signal acquisition methods follow a sample-then-compress procedure, CS could perform compression along with sampling. The novel acquisition strategy has enabled low-cost on-sensor data compression, relieving the pain for high transmission bandwidth and large storage space. In the recent decade, many algorithms have been proposed [3, 16, 1, 4, 22, 2, 12] to solve the CS reconstruction problem. Generally, these reconstruction algorithms are based on either optimization or greedy approaches using signal sparsity as prior knowledge. As a result, they all suffer from high computational complexity, which requires seconds to minutes to recover an image depending on the resolution. Therefore, these sparsity-based methods cannot satisfy the real-time decoding need of high-frame-rate cameras, and they are not appropriate for the high-frame-rate video CS

978-1-5386-4886-5/18/$31.00 ©2018 IEEE
DOI 10.1109/WACV.2018.00187

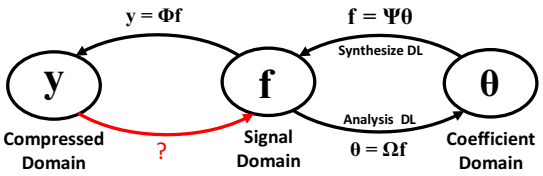

Figure 1: Illustration of domain transformations in CS. This work bridges the gap between compressed and signal domains.

application.

The slow reconstruction speed of conventional CS approaches motivates us to directly model the inverse mapping from the compressed domain to original domain, which is shown in Figure 1. Usually, this mapping is extremely complicated and difficult to model. However, the existence of massive unlabeled video data gives a chance to learn such a mapping using data-driven methods. In this paper, we design an enhanced Recurrent convolutional neural network (RCNN) to solve this problem. RCNN has shown astonishingly good performance for video recognition and description [5, 23, 25, 21]. However, conventional RCNNs are not well suited for video CS application, since they are mostly designed to extract discriminant features for classification related tasks. Simultaneously improving compression ratio (CR) and preserving visual details for high-fidelity reconstruction is a more challenging task. To solve this problem, we develop a special RCNN, called "CSVideoNet", to extract spatial-temporal features, including background, object details, and motions, to significantly improve the compression ratio and recovery quality trade-off for video CS application over existing approaches.

The contributions of this paper are summarized as follows:

- We propose an end-to-end and data-driven framework for video CS. The proposed network directly learns the inverse mapping from the compressed videos to the original input without additional pre/post-processing. To the best of our knowledge, there has been no published work that addresses this problem using similar methods.
- We propose a multi-level compression strategy to improve CR with the preservation of high-quality spatial resolution. Besides, we perform implicit motion estimation to improve temporal resolution. By combining both spatial and temporal features, we further improve the compression ratio and recovery quality trade-off without increasing much computational complexity.
- We demonstrate CSVideoNet outperforms the reference approaches not only in recovery quality but also in reconstruction speed because of its non-iterative nature. It enables real-time high-fidelity reconstruction for high-frame-rate videos at high CRs. We achieve state-of-the-art performance on the large-scale video dataset UCF-101. Specifically, CSVideoNet reconstructs videos at 125 fps on a Titan X GPU and achieves 25dB PSNR at a 100x CR.

2. Related work

There have been many recovery algorithms proposed for CS reconstruction, which can be categorized as follows:

Conventional Model-based CS Recovery: In [18], the authors model the evolution of scenes as a linear dynamical system (LDS). This model comprises two sub-models: the first is an observation model that models frames of video lying on a low-dimensional subspace; the second predicts the smoothly varied trajectory. The model performs well in stationary scenes, however, inadequate for non-stationary scenes.

In [27], the authors use Gaussian mixture model (GMM) to recover high-frame-rate videos, and the reconstruction can be efficiently computed as an analytical solution. The hallmark of the algorithm is that it adapts temporal compression rate based upon the complexity of the scene. The parameters in GMM are trained off-line and tuned during the recovery process.

In [19], the authors propose a multi-scale video recovery framework. It first obtains a low-resolution video preview with very low computational complexity, and then it exploits motion estimates to recover the full-resolution video by solving an optimization problem. In a similar work [8], the authors propose a motion-compensated and block-based CS reconstruction algorithm with smooth projected Landweber (MC-BCS-SPL). The motion vector is estimated from a reference and a reconstructed frame. The reconstructed video is derived from the combination of the low-resolution video and the estimated motion vector. The drawback of the two work is the requirement of specifying the resolution at which the preview frame is recovered, which requires prior knowledge of the object speed. Also, the recovery performance is highly dependent on the quality of motion estimation. To accurately estimate motion vector is a challenging task especially in high-frame-rate scenarios. The high computational cost further makes this model inadequate for reconstructing high-frame-rate videos.

Deep Neural Network (DNN) Based CS Recovery: In [15], the authors propose a stacked autoencoder to learn a representation of the training data and to recover test data from their sub-sampled measurements. Compared to the conventional iterative approaches, which usually need hundreds of iterations to converge, the feed-forward deep neural network runs much faster in the inference stage.

In [11], the authors propose a convolutional neural network, which takes CS measurements of an image as input and outputs an intermediate reconstruction. The intermediate output is fed into an off-the-shelf denoiser to obtain the final reconstructed image. The author shows the network is highly robust to sensor noise and can recover visually higher quality images than competitive algorithms at low CRs of 10 and 25. Both [15] and [11] are designed for image reconstruction, which only focus on spatial feature extraction. For video applications, temporal features between adjacent frames are also important. Therefore, the overlook of temporal correlation makes the image reconstruction algorithms inadequate for video applications.

In [9], the authors propose a Video CS reconstruction algorithm based on a fully-connected neural network. This work focuses on temporal CS where multiplexing occurs across the time dimension. A 3D volume is reconstructed from 2D measurements by a feed-forward process. The author claims the reconstruction time for each frame can be reduced to about one second. The major drawback of this work is that the algorithm is based on a plain fully-connected neural network, which is not efficient in extracting temporal features.

3. Methodology

3.1. Overview of the proposed framework for video CS

Two kinds of CS cameras are being used today. Spatial multiplexing cameras (SMC) take significantly fewer measurements than the number of pixels in the scene to be recovered. SMC has low spatial resolution and seeks to spatially super-resolve videos. In contrast, temporal multiplexing cameras (TMC) have a high spatial resolution but low frame-rate sensors. Due to the missing of inter frames, extra computation is needed for motion estimation. For these two sensing systems, either spatial or temporal resolution is sacrificed for achieving a better spatial-temporal trade-off. To solve this problem, we propose a new sensing and reconstruction framework, which combines the advantage of the two systems. The random video measurements are collected by SMC with very high temporal resolution. To compensate for the low spatial resolution problem in SMC, we propose a multi-CR strategy. The first *key frame* in a group of pictures (GOP) is compressed with a low CR, and the remaining *non-key frames* are compressed with a high CR. The spatial features in the key frame are reused for the recovery of the entire GOP due to the high inter-frame correlation in high-frame-rate videos. The spatial resolution is hence improved. The RNN extrapolates motion from high-resolution frames and uses it to improve the temporal resolution. Therefore, a better compression ratio and spatial-temporal resolution trade-off are obtained by the proposed framework.

The overall architecture of the proposed video CS reconstruction framework is shown in Figure 2. The network contains three modules: 1) an encoder (sensing matrix) for simultaneous sampling and compression; 2) a dedicated CNN for spatial features extraction after each compressed frame; 3) an LSTM for motion estimation and video reconstruction. As mentioned earlier, to improve the spatial resolution, the random encoder encodes the key frame in a GOP with more measurements and the remaining with less. Also, a recent research [26] shows that sensing matrix can be trained with raw data to better preserve the Restricted Isometry Property (RIP). Therefore, the encoder can also be integrated into the entire model and trained with the whole network to improve reconstruction performance. Besides, as the proposed algorithm eliminates the sparsity prior constraint, the direct optimization of RIP preservation in [26] is not necessary. Instead, we can use the reconstruction loss to train the sensing matrix along with the model. For simplicity, we still use a random Bernoulli matrix for information encoding in the experiment. Different from the prior work that extracts motion from low-resolution previews, the proposed LSTM network infers motion from high-resolution frames generated by multi-rate CNNs. The resolution of the reconstructed video is further improved with the incorporation of high-quality motion estimation.

3.1.1 Multi-rate CNN Encoder for compression ratio enhancement

Typical CNN architectures used for recognition, classification, and segmentation that map input to rich hierarchical visual features is not applicable to the reconstruction problem. The goal of the CNN is not only to extract spatial visual features but also to preserve details as much as possible. Therefore, we eliminated the pooling layer which causes information loss. Also, we discard the convolution-deconvolution architecture (widely used in segmentation tasks [17]), which first encodes salient visual features into low-dimension space and then interpolates the missing information to generate a high-resolution image. Instead, we design a special CNN suitable for CS reconstruction, which has the best recovery performance among all the tested structures mentioned above. The overall network structure is shown in Figure 3. All feature maps have the same dimension as the reconstructed video frames, and the number of feature maps decreases monotonically. This process resembles the sparse coding stage in CS, where a subset of dictionary atoms is combined to form the estimation of the original input. There is a fully-connected (FC) layer, denoted in gray color in Figure 3, which converts vectorized m-dimensional video data to 2D features maps. To reduce

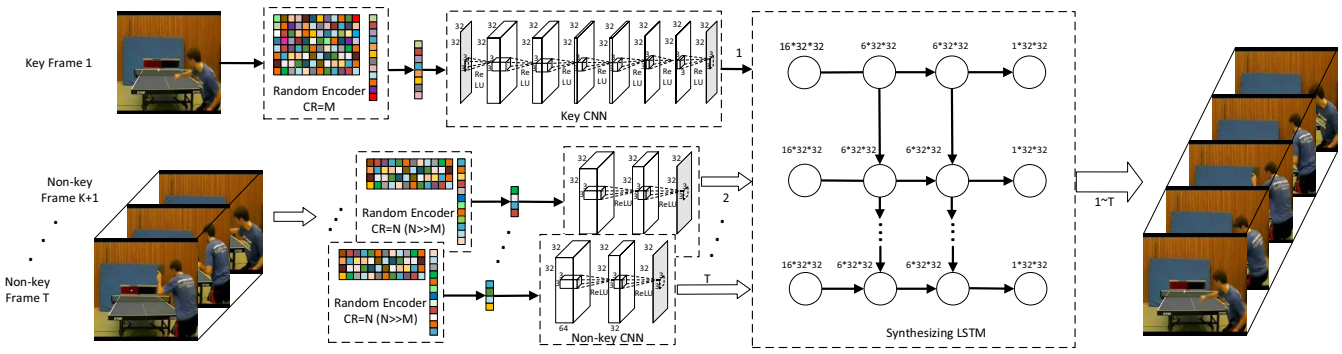

Figure 2: Overall architecture of the proposed framework. The compressed video frames are acquired by compressive sensing. In a length T GOP, the first one frame and the remaining (T-1) frames are compressed with a low and high CR, respectively. The reconstruction is performed by the CSVideoNet that is composed of a key CNN, multiple non-key CNNs, and a synthesizing LSTM.

the latency of the system and to simplify the network architecture, we use video blocks as input and set the block size n to 32×32. All convolutional layers are followed by a ReLU layer except the final layer. We pre-train an eight-layer *key CNN* to process the key frame that is compressed with a low CR. For other non-key frames compressed with a high CR, we use 3-layer *non-key CNNs* to handle them since they carry information of low entropy. All weights of the non-key CNNs are shared to reduce the requirement of storage. Hence the proposed framework can be easily generalized to other high-frame-rate video applications that require a larger number of non-key frames. It should be noted that the pre-training of the key CNN is critical for improving the reconstruction performance. In the case where the whole network is trained from scratch without any pre-training, the convergence performance is bad. The reason is partly due to the vanishing gradients, since we have a long path from the CNNs to the LSTM. The pre-training greatly alleviate this problem.

3.1.2 Motion-estimation synthesizing LSTM Decoder for spatial-temporal resolution enhancement

The proposed framework is end-to-end trainable, computationally efficient, and requires no pre/post-processing. This is achieved by performing motion estimation implicitly, which is different from prior works [19, 27, 8]. We utilize an LSTM network to extract motion features that are critical for improving temporal resolution from the CNN output. Since the information flows from the first LSTM node to the remaining, the LSTM will implicitly infers representations for the hidden motion from the key frame to the non-key frames. Therefore, the recovery quality of the GOP is improved by the aggregation of motion and spatial visual features. That is why we call this network the *motion-estimation synthesizing LSTM*. For simplicity, each input LSTM node in the experiment accepts input data with equal length. In fact, since the non-key frames carry less information than the key frame, the LSTM network can be designed to accept inputs with variable lengths. Hence, we can further reduce the model size and get a faster reconstruction speed. From the experiment results, we find the utilization of the LSTM network is critical to improving recovery fidelity. As a result, our model outperforms the competitive algorithms by a significant margin.

The update of the LSTM units is as follows:

$$\begin{aligned}
\mathbf{i}_t &= \sigma\left(W_{xi}\mathbf{x}_t + W_{hi}\mathbf{h}_{t-1} + W_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i\right), \\
\mathbf{f}_t &= \sigma\left(W_{xf}\mathbf{x}_t + W_{hf}\mathbf{h}_{t-1} + W_{cf}\mathbf{c}_{t-1} + \mathbf{b}_f\right), \\
\mathbf{c}_t &= \mathbf{f}_t\mathbf{c}_{t-1} + \mathbf{i}_t \tanh\left(W_{xc}\mathbf{x}_t + W_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c\right), \\
\mathbf{o}_t &= \sigma\left(W_{xo}\mathbf{x}_t + W_{ho}\mathbf{h}_{t-1} + W_{co}\mathbf{c}_t + \mathbf{b}_o\right), \\
\mathbf{h}_t &= \mathbf{o}_t \tanh(\mathbf{c}_t),
\end{aligned}$$

where $\mathbf{x}_t$ is the visual feature output of the CNN encoder. The detailed information flow and the output dimension at each LSTM node is shown in Figure 2. The number on the LSTM nodes denotes the dimension of the output features. Specifically, the output feature map of each CNN has a dimension of 16x32x32. All these feature maps are directly fed into the input nodes of the LSTM. The LSTM has two hidden layers, the dimension of the output of each hidden layer is 6x32x32. The dimension of the final output is 1x32x32.

3.2. Learning algorithm

Given the ground-truth video frames $x_{\{1,\cdots,T\}}$ and the corresponding compressed frames $y_{\{1,\cdots,T\}}$, we use mean square error (MSE) as the loss function, which is defined as:

$$L(\mathbf{W}, \mathbf{b}) = \frac{1}{2N}\sum_{i}^{T} \|f(y_i; \mathbf{W}, \mathbf{b}) - x_i\|_2^2, \qquad (1)$$

where $\mathbf{W}$, $\mathbf{b}$ are network weights and biases, respectively.

Using MSE as the loss function favors high PSNR. PSNR is a commonly used metric to quantitatively evaluate recovery quality. From the experiment results, we illustrate that PSNR is partially correlated to the perceptual quality. To derive a better perceptual similarity metric will be a future work. The proposed framework can be easily adapted to a new loss function.

Three training algorithms, i.e., SGD, Adagrad [7] and Adam [10] are compared in the experiment. Although consuming most GPU memory, Adam converges towards the best reconstruction results. Therefore, Adam is chosen to optimize the proposed network.

4. Experiment

As there is no standard dataset designed for video CS, we use UCF-101 dataset introduced in [20] to benchmark the proposed framework. This dataset consists of 13k clips and 27 hours of video recording data collected from YouTube, which belong to 101 action classes. Videos in the dataset are randomly split into 80% for training, 10% for validation and the remaining for testing. Videos in the dataset have a resolution of 320×240 and are sampled at 25 fps. We retain only the luminance component of the extracted frames and crop the central 160×160 patch from each frame. These patches are then segmented into 32×32 non-overlapping image blocks. We get 499,760 GOPs for training and testing in total.

We set three test cases with CRs of 25, 50 and 100, respectively. Since the CR for key and non-key frames are different in the proposed method, we derive and define the CR for a particular GOP as follows. Let $m1, m2$ denotes the dimension of compressed key and non-key frame, respectively. Let n denotes the dimension of raw frames. T is the sequential length of a GOP.

$$\begin{aligned} CR_1 &= n/m1, CR_2 = n/m2, \\ CR &= \frac{CR_1 \times 1 + CR_2 \times (T-1)}{T}. \end{aligned} \tag{2}$$

In the experiment, the CR of each key frame is m1=5, and the CR of non-key frames in each test case is m2=27, 55, and 110, respectively. Therefore, the averaged CR for each test case is about 25, 50, and 100, respectively.

The dimension of data for pre-training the key CNN is $(N \times C \times H \times W)$, where N=100 is the batch size, C=1 is the channel size, and W, H=(32, 32) is the height and width of each image block, respectively. The dimension of the data used for training the entire model is $(N' \times T \times C \times H \times W)$, where T=10 is the sequence length for one GOP, and N'=20 is the batch size. The other dimensions are the same. We shrink the batch size here because of the GPU memory limitation. In every ten consecutive video frames, we define the first one as the key frame, and the remaining as non-key frames.

Table 1: Summary of major differences between the proposed approach and all baselines.

Image CS	Iterative Based	Denoising-based approximate message passing	D-AMP [14]
	Non-iterative Based	Stacked denoising autoencoder	SDA [15]
		Convolutional neural network	ReconNet [11]
Video CS	Iterative Based	Motion-compensated block-based CS with smooth projected Landweber	MC-BCS-SPL [8]
		Gaussian mixture model	GMM [27]
	Non-iterative Based	Fully-connected neural network	VCSNet [9]
		Proposed approach	**CSVideoNet**

4.1. Comparison with the state-of-the-art

We compare our algorithm with six reference work for CS reconstruction: [27, 8, 15, 14, 11, 9]. We summarize all baseline approaches and our approach in Table 1. For a fair comparison, we also re-train reference algorithms using UCF-101 dataset. Three metrics: Peak signal-to-noise ratio (PSNR), structural similarity (SSIM) [24], and pixel-wise mean absolute error (MAE) are applied for performance evaluation. Note that MAE is the averaged absolute error of each pixel value within the range of [0,255], which gives a straightforward measure of the pixel-wise distortion. The authors of VCSNet only offer a pre-trained model with CR of 16, without providing sufficient training details to reproduce the experiment at present. Therefore, we train the proposed model and compare it with CVSNet at a single CR of 16.

4.1.1 Comparison with image CS approaches

We first compare with the algorithms used for image CS reconstruction. D-AMP is a representative of the conventional iterative algorithms developed for CS, e.g., matching pursuit, orthogonal mating pursuit, iterative hard-thresholding. It offers state-of-the-art recovery performance and operates tens of times faster compared to other iterative methods [14]. Both SDA and ReconNet are DNN-based reconstruction approaches for images proposed recently. Specifically, ReconNet is based on CNN and achieves state-of-the-art performance among all image CS reconstruction algorithms [11]. In the experiment, we tested both frame-based and block-based D-AMP that reconstructs an entire frame and an image block at a time, respectively. For other approaches, we test them in a block-based pattern to reduce the difficulty for training the models. The quantized results of average PSNR, SSIM, and MAE for each method under different CRs are shown in Table 2. It is shown that CSVideoNet outperforms the reference approaches on all three metrics by a meaningful margin, especially at the CR of 100. The MAE of CSVideoNet is 4.59 at a 100x CR which means the averaged pixel-wise distortion is only $4.59/255 = 1.2\%$ compared to the ground-truth video. The PSNR drop from the CR of 25 to 100 is also cal-

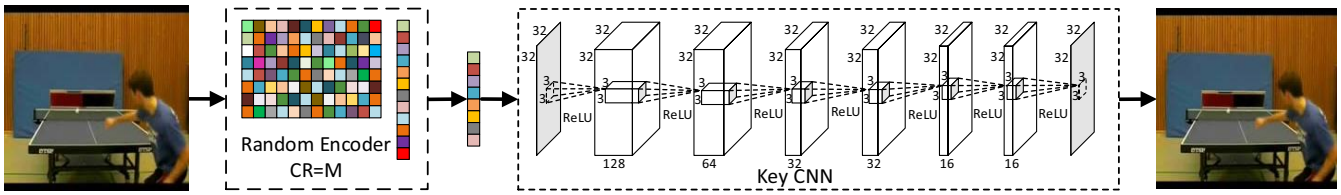

Figure 3: Pre-training of the key CNN.

Table 2: Performance comparison with image CS reconstruction approaches.

	CR	D-AMP(F)	D-AMP(B)	SDA	ReconNet	CSVideoNet
PSNR	25	25.34	15.1494	23.39	24.27	**26.87**
	50	12.49	9.1719	21.96	22.47	**25.09**
	100	7.17	8.0942	20.40	20.44	**24.23**
SSIM	25	0.76	0.0934	0.69	0.73	**0.81**
	50	0.08	0.0249	0.65	0.67	**0.77**
	100	0.03	0.0067	0.61	0.61	**0.74**
MAE	25	4.65	24.92	5.76	5.02	**3.38**
	50	64.30	81.67	6.60	5.67	**4.31**
	100	92.12	86.04	8.50	7.42	**4.59**
PSNR↓	25 → 100	72%	13%	47%	16%	**10%**

culated in Table 2. We found the proposed approach suffers from the least performance degradation. This is partly due to the feature sharing between the key and non-key frames when the compressed input carries limited information.

For visual quality assessment purpose, we list the reconstructed frame by each approach in Figure 4. The reconstructed frame is the middle (fifth) frame in a GOP. We find all the reconstructed non-key frames have homogeneous recovery quality, and the key frame has slightly better reconstruction quality than the non-key frames. As the proportion of key and non-key frames is 1:9, and the reconstruction quality of the video is dominated by that of the non-key frames. Therefore, the middle frame (a non-key frame) shown in Figure 4 well represents the average reconstruction quality.

For all the numerical results, we calculate all the quality metrics, including PSNR, SSIM, and MAE, by averaging the results over all frames in a GOP. We can see that CSVideoNet provides the finest details among all approaches. The edges produced by CSVideoNet is much sharper, while such details are no longer preserved by other methods after reconstruction. This comparison demonstrates that the temporal correlation is critical for video reconstruction, the overlook of such features will significantly degrade the recovery quality of videos. Therefore, the conventional image CS approaches are not suitable for video applications.

4.2. Comparison with video CS approaches

We compare the proposed CSVideoNet with existing video CS approaches. MC-BCS-SPL estimates motion directly from the current and the reference frame. GMM models the spatial-temporal correlation by assuming all

Table 3: Performance comparison with video CS reconstruction approaches.

	CR	MC-BCS-SPL	GMM	CSVideoNet
PSNR	25	22.41	23.76	**26.87**
	50	20.59	21.26	**25.09**
	100	19.67	19.64	**24.23**
SSIM	25	0.37	0.72	**0.81**
	50	0.30	0.61	**0.77**
	100	0.19	0.54	**0.74**
MAE	25	11.88	5.14	**3.38**
	50	16.03	7.50	**4.31**
	100	28.86	9.37	**4.59**
PSNR↓	25 → 100	26%	17%	**10%**

pixels within a video patch are drawn from a GMM distribution. GMM has the state-of-the-art performance among conventional model-based video CS approaches [27]. To the best of our knowledge, [9] is the only DNN-based work proposed for video CS. The quantized results of average PSNR, SSIM, and MAE for each method under different CRs are shown in Table 3. It is observed that the proposed approach improves PSNR by 3 to 5dB over the reference methods. Specifically, we find MC-BCS-SPL and GMM have similar performance and perform much better than the model-based image CS approach, D-AMP. However, their performance are similar to SDA and ReconNet, which are designed for processing images. This implies that the conventional model-based methods suffer from limited performance due to the limited model capacity when dealing with large-scale problem. Even though they consider the temporal correlation among video frames, the model capacity is insufficient for visual patterns. To improve performance, one could increase the size of the conventional models. However, the computational complexity forof these meods will also increase substantially, inhibiting their application to video CS.

DNN provides a viable solution. Both CSVideoNet and VCSNet are designed for video CS reconstruction. For reasons explained earlier, we compare the two approaches at a CR of 16. The results are shown in Table 4 and Figure 5. Both the two approaches achieve high recovery quality compared to other baselines. However, VCSNet is a plain fully-connect network that has limited capability for processing sequential data. As a result, it suffers from a low-quality motion estimation, which explains why it has inferior performance compared to the proposed solution.

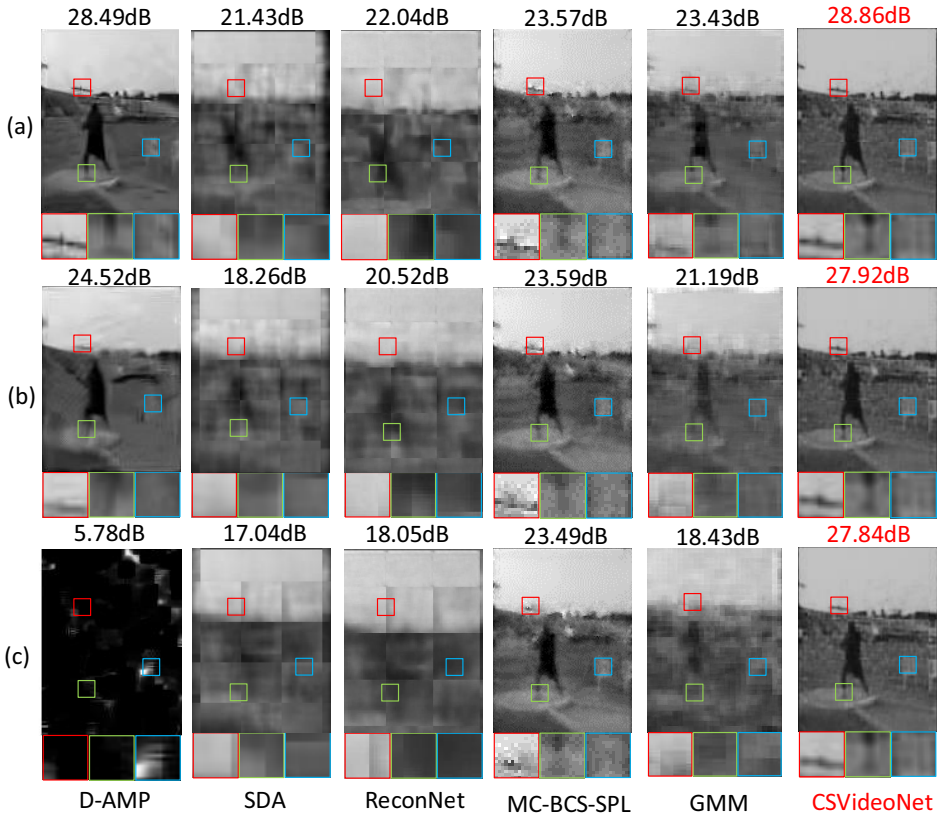

Figure 4: Illustration of reconstruction results for each method at the CR of (a) 25, (b) 50, and (c) 100, respectively.

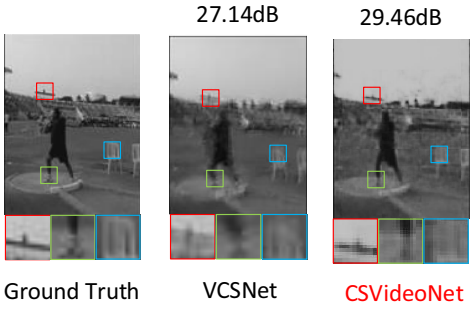

Figure 5: Illustration of reconstruction results at the CR of 16.

Table 4: Performance comparison with VCSNet at the CR of 16.

	VCSNet	CSVideoNet
PSNR	25.07704	28.078
SSIM	0.817669	0.8431
MAE	3.887867	2.9452

To illustrate that the performance improvement of the proposed approach comes from integrating temporal features through the LSTM network rather than simply increasing the model size, we set another experiment, in which we compare the performance of two CNNs with different sizes. The structure of the two CNNs are shown in Table 5, and the performance comparison is shown in Table 7. We can see that simply increasing the size of CNN does not provide meaningful improvement for reconstruction. This, wh be explained by the incapability of CNN to capture temporal features. The incorporation of the LSTM network improves the PSNR by up to 4 dB, which represents more than twice of error reduction. Specifically, the performance improvement increases with thealong wiachieves theits maximum wheR is 100. This explains that the implicit motion estimation by LSTM is critical to the video CS reconstruction especially at high CRs.

Table 5: Structures of CNN1 and CNN2.

# Layer	1	2	3	4	5	6	7	8	9	10	11	12	13
CNN1	1	128	64	32	32	16	16	1					
CNN2	1	512	256	256	128	128	64	64	32	32	16	16	1

* CNN1 is used in CSVideoNet. The dimension of all feature maps in both CNNs are 32×32.

4.3. Performance under noise

To demonstrate that the robustness of CSVideoNet to sensor noise, we conduct a reconstruction experiment with input videos contaminated by random Gaussian noise. In this experiment, the architecture of all DNN-based frameworks remains the same as in the noiseless case. We test the performance at three levels of SNR - 20dB, 40dB, and

Table 6: Runtime comparison for reconstructing a 160×160 video frame at different CRs.

Model	CR=25	CR=50	CR=100
D-AMP(F)	38.37	41.20	31.74
D-AMP(B)	8.4652	8.5498	8.4433
SDA	0.0278	0.027	0.023
ReconNet	0.064	0.063	0.061
MC-BCS	7.17	8.03	9.00
GMM	8.87	10.54	18.34
CSVideoNet	0.0094	0.0085	0.0080

Table 7: Performance comparison with CNN methods.

	CR	CNN1	CNN2	CSVideoNet
PSNR	25	24.27	23.74	26.87
	50	22.47	22.17	25.09
	100	20.44	20.10	24.23
SSIM	25	0.73	0.69	0.81
	50	0.67	0.65	0.77
	100	0.61	0.58	0.74
MAE	25	5.02	6.46	3.38
	50	5.67	6.23	4.31
	100	7.42	8.92	4.59

60dB. For each noise level, we evaluate all approaches at three CRs of 25, 50, and 100. The average PSNR achieved by each method at different CRs and noise levels are shown in Figure 6. It can be observed that CSVideoNet can reliably achieve a high PSNR across at different noise levels and outperform the reference methods consistently.

4.4. Time complexity

We benchmark the runtime performance of different methods. Due to the iterative nature of conventional CS algorithms (D-AMP, MC-BCS-SPL, GMM), they suffer from high data-dependency and low parallelism, which is not suitable for GPU acceleration. Due to the lack of GPU solvers, we run these reference algorithms on an octa-core Intel Xeon E5-2600 CPU. Benefiting from the feedforward data-path and high data concurrency of DNN-based approaches, we accelerate CSVideoNet and other DNN-based baselines using a Nvidia GTX Titan X GPU. The time cost for fully reconstructing a video frame in the size of (160×160) are compared in Table 6. CSVideoNet consumes 8 milliseconds (125 fps) to reconstruct a frame at the CR of 100. This is three orders of magnitude faster than the reference methods based on iterative approaches. The time cost of VCSNet and CSVideoNet at the CR of 16 is 3.5 and 9.7 milliseconds, respectively. Through further hardware optimization, we believe CSVideoNet has the potential to be integrated into CS cameras to enable the real-time reconstruction of high-frame-rate video CS.

5. Conclusion

In this paper, we present a real-time, end-to-end, and non-iterative framework for high-frame-rate video CS. A multi-rate CNN variant and a synthesizing LSTM network are developed to jointly extract spatial-temporal features. This is the key to enhancing the compression ratio and recovery quality trade-off. The magnificent model capacity of the proposed deep neural network allows to map the inverse mapping of CS without exploiting any sparsity constraint. The feed-forward and high-data-concurrency natures of the proposed framework are the key to enabling GPU acceleration for real-time reconstruction. Through performance comparison, we demonstrate that CSVideoNet has the potential to be extended as a general encoding-decoding framework for high-frame-rate video CS applications. In the future work, we will exploit the effective learning methods to decode high-level information from compressed videos, e.g., object detection, action recognization, and scene segmentation.

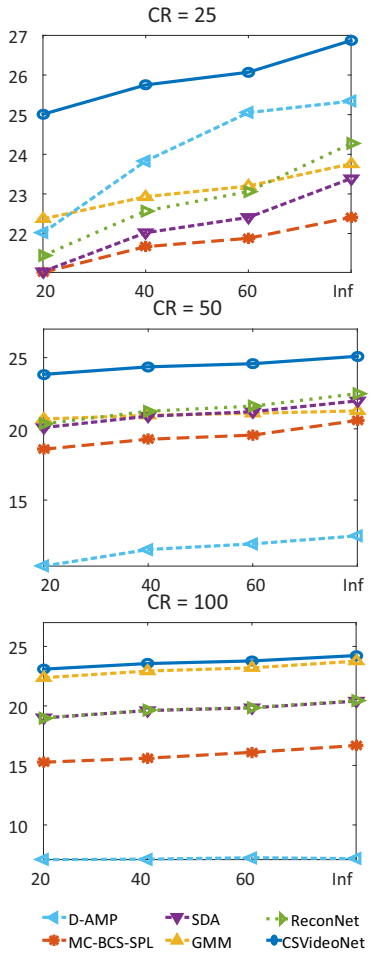

Figure 6: PSNR comparison at different SNRs.

6. Acknowledgement

This work is supported by NSF grant IIS/CPS-1652038. The research infrastructure used by this work is supported by NSF grant CNS-1629888.

References

[1] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. 1

[2] T. Blumensath and M. E. Davies. Iterative hard thresholding for compressed sensing. *Applied and Computational Harmonic Analysis*, 27(3):265 – 274, 2009. 1

[3] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE TIT*, 52(2):489–509, Feb 2006. 1

[4] I. Daubechies, R. DeVore, M. Fornasier, and C. S. Gntrk. Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics*, 63(1):1–38, 2010. 1

[5] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015. 2

[6] M. F. Duarte, M. A. Davenport, D. Takbar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 2008. 1

[7] J. C. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12:2121–2159, 2011. 5

[8] J. E. Fowler, S. Mun, and E. W. Tramel. Block-based compressed sensing of images and video. *Foundations and Trends in Signal Processing*, 4(4):297–416, 2012. 2, 4, 5

[9] M. Iliadis, L. Spinoulas, and A. K. Katsaggelos. Deep fully-connected networks for video compressive sensing. *Digital Signal Processing*, 72(Supplement C):9 – 18, 2018. 3, 5, 6

[10] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2015. 5

[11] K. Kulkarni, S. Lohit, P. Turaga, R. Kerviche, and A. Ashok. Reconnet: Non-iterative reconstruction of images from compressively sensed measurements. In *CVPR*, 2016. 3, 5

[12] D. Li, X. Wang, and D. Kong. DeepRebirth: Accelerating Deep Neural Network Execution on Mobile Devices. In *AAAI*, 2017. 1

[13] S. Ma, W. Yin, Y. Zhang, and A. Chakraborty. An efficient algorithm for compressed mr imaging using total variation and wavelets. In *CVPR*, 2008. 1

[14] C. A. Metzler, A. Maleki, and R. G. Baraniuk. From denoising to compressed sensing. *IEEE TIT*, 62(9):5117–5144, Sept 2016. 5

[15] A. Mousavi et al. A Deep Learning Approach to Structured Signal Recovery. *CoRR*, abs/1508.04065, 2015. 2, 3, 5

[16] D. Needell and J. A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *ACM Communications*, 53(12):93–100, 2010. 1

[17] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *ICCV*, 2015. 3

[18] A. Sankaranarayanan, P. Turaga, R. Chellappa, and R. Baraniuk. Compressive acquisition of linear dynamical systems. *SIAM Journal on Imaging Sciences*, 6(4):2109–2133, 2013. 2

[19] A. C. Sankaranarayanan, L. Xu, C. Studer, Y. Li, K. F. Kelly, and R. G. Baraniuk. Video compressive sensing for spatial multiplexing cameras using motion-flow models. *SIAM Journal on Imaging Sciences*, 8(3):1489–1518, 2015. 2, 4

[20] K. Soomro, A. Roshan Zamir, and M. Shah. UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild. In *CRCV-TR-12-01*, 2012. 5

[21] N. Srivastava, E. Mansimov, and R. Salakhudinov. Unsupervised learning of video representations using lstms. In *ICML*, 2015. 2

[22] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE TIT*, 53(12):4655–4666, 2007. 1

[23] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. Sequence to sequence - video to text. In *ICCV*, 2015. 2

[24] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE TIP*, 13(4):600–612, April 2004. 5

[25] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015. 2

[26] K. Xu, Y. Li, and F. Ren. A Data-Driven Compressive Sensing Framework Tailored For Energy-Efficient Wearable Sensing. In *ICASSP*, 2017. 3

[27] J. Yang, X. Yuan, X. Liao, P. Llull, D. J. Brady, G. Sapiro, and L. Carin. Video compressive sensing using gaussian mixture models. *IEEE TIP*, 23(11):4863–4878, 2014. 2, 4, 5, 6