CSVideoNet: A Recurrent Convolutional Neural Network for Compressive Sensing Video Reconstruction

Kai XU Fengbo Ren

Abstract

In this paper, we develop a deep neural network architecture called "CSVideoNet" that can learn visual representations from random measurements for compressive sensing (CS) video reconstruction. CSVideoNet is an end-to-end trainable and non-iterative model that combines convolutional neural networks (CNNs) with a recurrent neural networks (RNN) to facilitate video reconstruction by leveraging temporal-spatial features. The proposed network can accept random measurements with a multi-level compression ratio (CR). The lightly and aggressively compressed measurements offer background information and object details, respectively. This is similar to the variable bit rate techniques widely used in conventional video coding approaches. The RNN employed by CSVideoNet can leverage temporal coherence that exists in adjacent video frames to extrapolate motion features and merge them with spatial visual features extracted by the CNNs to further enhance reconstruction quality, especially at high CRs. We test our CSVideoNet on the UCF-101 dataset. Experimental results show that CSVideoNet outperforms the existing video CS reconstruction approaches. The results demonstrate that our method can preserve relatively excellent visual details from original videos even at a 100x CR, which is difficult to realize with the reference approaches. Also, the non-iterative nature of CSVideoNet results in an decrease in runtime by three orders of magnitude over iterative reconstruction algorithms. Furthermore, CSVideoNet can enhance the CR of CS cameras beyond the limitation of conventional approaches, ensuring a reduction in bandwidth for data transmission. These benefits are especially favorable to high-frame-rate video applications.

1. Introduction

High-speed cameras are capable of capturing video at frame rates more than one hundred frames per second (fps). These devices were originally developed for research purpose, e.g., physical and biological science, to characterize events which occur at a rate that is higher than traditional cameras are capable of recording. High-end, high-speed cameras, such as Photron SA1, SA3, are capable of recording high resolution still images of extremely high-speed events such as a supersonic flying bullet or an exploding balloon with negligible motion blur and image distortion artifacts. However, due to the complex architecture of high-speed cameras, the costs of these types of equipment are extremely financially expensive; usually over tens of thousand dollars for each, rendering the devices to be considered to be too expensive for the consumer market. Furthermore, the challenges associated with transmission and storage also contribute to the inhibiting practical manufacture of affordable consumer devices that use this technology. For examples, true high definition video resolution, e.g. 1080p, cameras at a frame rate of 10kfps can generate about 500GB of data per second, which poses significant challenges for existing transmission and storing techniques. Recently, companies have begun to manufacture high-speed cameras in commercial products, for example. For example, "GoPro 5" can capture videos at 120 fps with 1080p resolution. However, the short battery life, which is approximately 1-2 hours, has significantly limited their range of practical fields of application.

Compressive sensing (CS) is an acquisition technique that allows for full reconstruction with a sub-Nyquist sampling rate. The advent of CS has led to the emergence of new image devices, e.g., single-pixel cameras [12] and has also been applied in practical applications, e.g., accelerated magnetic resonance imaging (MRI) [24]. Whereas traditional signal acquisition methods first sample a signal followed by compressing it, CS performs the compression of the signal during the acquisition stage. This alternative to sampling at the Nyquist rate reduces the amount of data to record which gives rise to a lower cost of both sampling and storage. In CS, the challenge usually lies in the reconstruction stage, because we have minimal prior knowledge of the input data that is of interest. In recent decades, many reconstruction algorithms have been developed [6, 27, 3, 10, 36, 4] to solve this problem. Generally, reconstruction algorithms are based either on optimization or greedy iteration. These methods all suffer from high complexity in terms of computation, which can cause the reconstruction to consume up to 10 minutes to recover an image. For high-speed video applications, obviously, these methods are not practical.

The recent development of deep learning technique have proven that Convolutional Neural Networks (CNNs) are good at capturing visual features for classification, detection, recognition, and segmentation [19, 20, 13, 29, 22, 1]. Another type of neural network that has proven successful in processing sequences with temporal or sequential structure are Recurrent Neural Networks (RNNs), which are now widely used in nature language processing (NLP) tasks, such as speech recognition [16, 14], machine translation [2, 8], caption generation [38, 7], and visual question answering [28, 40]. For video reconstruction applications, it is essential for the learner to model the temporal progression [35]. RNNs allow for temporal information to be utilized in the reconstruction process by constructing the current frame based on the information contained in both the current frame and extrapolated temporal dependencies among patches. This method of reconstruction is in contrast to the traditional technique of treating video as a set of independent images. This property of using RNNs to include temporal information into the reconstruction process can be exploited to produce more accurate models.

In this work, we utilize both CNNs and RNNs to extract spatial-temporal features, including background, object details and motion, achieving a better reconstruction quality for video CS application compared to existing algorithms.

The contributions of this paper are summarized as follows:

- We propose a long-term recurrent convolutional network (LRCN) based algorithm for video reconstruction from compressed measurements. To the best of our knowledge, there is no published work addressing this problem using similar methods.

- The proposed model can accept measurements with a multi-level compression ratio (CR). Generally, the first few frames in a short video series (referred to as key frames) can be considered to carry more background information than the rest frames (referred to as non-key frames) that mostly contribute motion information. In our framework, the key and non-key frames are lightly and aggressively compressed, respectively. The RNN can extrapolate the motion information and combine it with the visual features extracted by the CNNs to synthesize high-quality frames. The efficient information fusion enables a better trade-off between fidelity and CR for CS video applications.

- We achieve state-of-the-art reconstruction performance on the large-scale video dataset UCF-101 [34].

- We show the proposed model outperforms the competitor algorithms in terms of reconstruction speed, and demonstrate it can be used for real-time high-speed video CS reconstruction.

2. Related Works

Conventional Model Based Video Compressive Sensing In [31], the author models the evolution of the scenes as a linear dynamical system (LDS), which comprises two sub-models: the first is an observation model that models frames of video that lie close to 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 [39], the author use Gaussian mixture model (GMM) as a signal prior to recovery high-speed videos, and the reconstruction can be efficiently computed by an analytical solution. The hallmark of the algorithm is that it adapts rate of temporal compression and utilizes the reconstructed information to learn a multi-class classifier for changing the temporal length for each patch. The GMM parameters are trained offline and tuned during the recovery process.

In [32], the author proposes a multi-scale video recovery framework. It first obtains a low-resolution video preview with very low computational complexity, then it exploits motion estimates to recover the full-resolution video by an optimization algorithm. In a similar work [26], the author proposes a motion-compensated and block-based CS (MC-BCS) reconstruction algorithm which also estimate the motion information. By utilizing the combination of the low-resolution video and motion information, the reconstructed video can be improved substantially. The drawback of this approach is the need to specify the resolution at which the preview frame is recovered, which requires prior knowledge of object speed. Furthermore, the extraction of motion information has a high runtime cost which makes this model inadequate for reconstructing high-speed videos.

Auto-encoder based CS recovery: In [17], the author proposes that one can reduce the dimensionality of the data and reconstruct the original data via an autoencoder, which demonstrates the potential of the deep networks for information reconstruction. Based on that, in [25], the author uses a stacked autoencoder to learn a representation of the training data and recovers test data from their sub-sampled measurements. Compared to the conventional convex optimization based approaches, which usually need hundreds of iterations to converge, the feed-forward deep neural network is much faster at the inference stage.

CNN based CS recovery: In [21], the author proposes 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 extremely low CRs of 10 and 25. The author shows the proposed algorithm outperforms the competing algorithms by considerable margins and presents a proof of concept real-time application, wherein object tracking is performed on the fly as the frames are recovered from the CS measurements. [25] and [21] are designed for image reconstruction, and they all focus on spatial features that exist within frames. For video applications, the temporal features between frames are equally or even more important than spatial features. The neglect of temporal information makes the image reconstruction algorithms inadequate for video reconstruction applications.

In [18] [1], the author proposed a video CS reconstruction algorithm based on a fully-connected neural network, that learns a mapping that maps temporal CS measurements to video frames. This work focuses on temporal CS where multiplexing occurs across the time dimension, a 3D volume is reconstructed from 2D measurements after a feed-forward process. The author claims the reconstruction time for each frame can be reduced to a few seconds. The major drawback of this work is that fully-connected neural networks are not efficient in extracting temporal features.

3. Overview of Our Algorithm

In this section, we introduce the challenges behind video CS and propose an algorithm called "CSVideoNet" for solving the problem.

3.1. Challenges in Video CS

Assuming a signal $\mathbf{f}$ can be represented by a sparse vector $\theta \in \mathbb{R}^k$ on a certain basis $\Psi \in \mathbb{R}^{n\times k}$, i.e., $\mathbf{f} = \Psi\theta$, the signal information $\mathbf{x}$ can be well preserved by projecting $\mathbf{f}$ onto a low-dimensional space through a sensing matrix $\Phi \in \mathbb{R}^{m\times n}$, $(m \leq n)$, given as

$$\mathbf{y} = \Phi\mathbf{f} = \Phi\Psi\theta. \tag{1}$$

For robust reconstruction, the sensing matrix Φ should satisfy the Restricted Isometry Property (RIP) [5] of 2K order for all K-sparse signal pairs $\{\mathbf{x_1}, \mathbf{x_2}\}$, defined as

$$1 - \delta_{2K} \leq \frac{\|\boldsymbol{\Phi}(\mathbf{x_1} - \mathbf{x_2})\|_2}{\|(\mathbf{x_1} - \mathbf{x_2})\|_2} \leq 1 + \delta_{2K}, \tag{2}$$

where δ_{2K} is the isometry constant. When $\delta_{2K} < \sqrt{2} - 1$, Φ approximately preserves the Euclidean norm of all K-sparse signals and guarantees the existence and uniqueness of the reconstruction. The sparse coefficient can be solved via the following ℓ-1 minimization,

$$\min_{\theta \in \mathbb{R}^k} \|\theta\|_1 \quad s.t. \quad \mathbf{y} = \boldsymbol{\Phi\Psi}\theta, \tag{3}$$

where $\boldsymbol{\Psi}$ is the dictionary, θ is the sparse coefficient and $\mathbf{y}$ is the compressed measurement. The original input can be reconstructed by calculating $\mathbf{x} = \boldsymbol{\Psi}\theta$ and this is the fundamental theory behinds conventional CS measurement systems. However an optimization based solution for problem 3 is time-consuming [30], a new algorithm with low computational complexity is expected for high-speed video reconstruction. Equation 1 shows what we want to learn is an inversion Φ^{-1} which maps the compressed measurements back to the original domain. With a sufficient amount of data, a deep neural network can learn sophisticated models. That motivates us to utilize deep neural networks to determine the inverse mapping.

Another challenge in videos reconstruction is that the temporal dimension is fundamentally different from the spatial dimension due to its ephemeral nature. The causality of time prevents us from obtaining additional measurements of an event that has already occurred. The traditional methods that aggregate multiple measurements over time to recover the video via sparse recovery exhibits poor performance. [32]. This algorithm relies on estimating the motion in the scenes, then compress a few reference frames, and use the motion that relate the remaining parts of a scene to these reference frames. This is not possible in compressive sensing since we have not accessed the video before compression. The general strategy of motion flow based models for video CS is first to recover each frame individually and then derive motion flow between consecutive frames. The video is estimated with the aid of enforcing the extracted motion-flow constraints. Unfortunately,

[1]This work is submitted to arXiv.org, not peer reviewed and published yet.

the method highly depends on the quality of the derived motion-flow, which is hard to obtain due to the violation of the static scene model. [32].

To solve these problems, we propose to use CNN and LSTM together for video reconstruction. CNNs are efficient in learning the hierarchical representations within each frame by utilizing the convolution operation. RNN can learn hidden information that is needed to extrapolate motion and appearance beyond what has been observed, and use the motion as well as the static visual features to build a high-quality reconstruction. We believe the CR can be promoted by the utilization of spatial-temporal coherence within and between frames in videos. Similar architecture called long-term recurrent convolutional networks have already been successfully applied for visual description [11, 41] and activity recognition [23, 33], and termed as long-term recurrent convolutional networks (LRCNs). We have enhanced this model and devised a new one that is particularly suitable for video CS reconstruction.

3.2. Network Architecture

The overall architecture of the proposed algorithm is shown in Figure 1. The network contains three modules: a random encoder for measurement, a CNN cluster for visual feature extraction and an LSTM for temporal reconstruction. The random encoder works in a parallel manner, encoding the first frame in a video patch with more measurements, and the remaining frames with less measurements. For each compressed measurement, there is a dedicated CNN to extract spatial features from it. The LSTM network aggregates all the features extracted by each CNN, along with the inferred motion from hidden states to form the reconstruction. More details of the proposed network are discussed in the following subsections.

3.2.1 CNN Design

Typical CNN architectures used for recognition, classification and segmentation that maps input to rich hierarchical visual features are not applicable to the reconstruction problem. Therefore, we designed a new CNN that takes compressed measurements and outputs reconstructions for images, shown in Figure 3. To get a higher CR, we separate each video patch which contains T frames into K key frames and (T-K) non-key frames. The key frames are compressed with low CR and non-key frames with high CR. We hope the measurement information of key frames can be reused in the reconstruction of non-key frames, which can be viewed as temporal compression. Therefore, our model combines both temporal and spatial compression to maximize CR. We design a larger CNN to handle key frames since it contains information with high entropy, and a smaller CNN to deal with non-key frames. To reduce the latency of the system as well as simplifying the network architecture, we use image blocks as input and the size of all feature maps generated by the CNNs are the same as the image blocks, the number of feature maps are decreased monotonically. The input to the network is an m-dimensional vector composed of compressed measurements. There is a fully-connected layer before CNN that takes these measurements and generates a 2-D feature map.

3.2.2 LSTM Design

To obtain an end-to-end trainable and computationally efficient model, we perform no pre-processing to the original input. Also, we do not estimate the optical flow of videos explicitly, instead we utilize an LSTM network to extract the motion features that are essential for reconstruction. In the proposed model, the synthesizing LSTM network is used for motion extrapolation and aggregation of spatial visual features and motion for reconstruction. The training process of LSTM network is shown in Figure 4. The first M inputs of LSTM takes the data from the CNNs dealing with key frames, and the remaining (T-M) takes the outputs from the CNNs dealing with non-key frames. For each LSTM units, it will receive visual features from the key frames, which are used for background reconstruction, current frame for recover objects and the last few frames for motion estimation. From the experiment results, we find the utilization of the LSTM network is critical to improve recovery fidelity and our model outperforms the competitive algorithms by a significant margin.

The output of the proposed model still contains some artifacts and noise; various denoising algorithms are proposed to remove them, e.g., [9, 37, 15]. We believe the denoising module will improve the reconstruction performance. However, this is not the focus of our work, so we did not introduce any denoising module in the experiment.

3.3. Learning Algorithm

We divide the network training process into two stages. In the first stage, we pre-train the background CNN and extract visual features from K key frames, shown in Figure 3. In the second stage, to give the model more freedom to extract “essential” blocks for building objects, we train (T-M) small CNNs from scratch. These object CNNs and the pre-trained background CNNs are incorporated with a synthesiz-

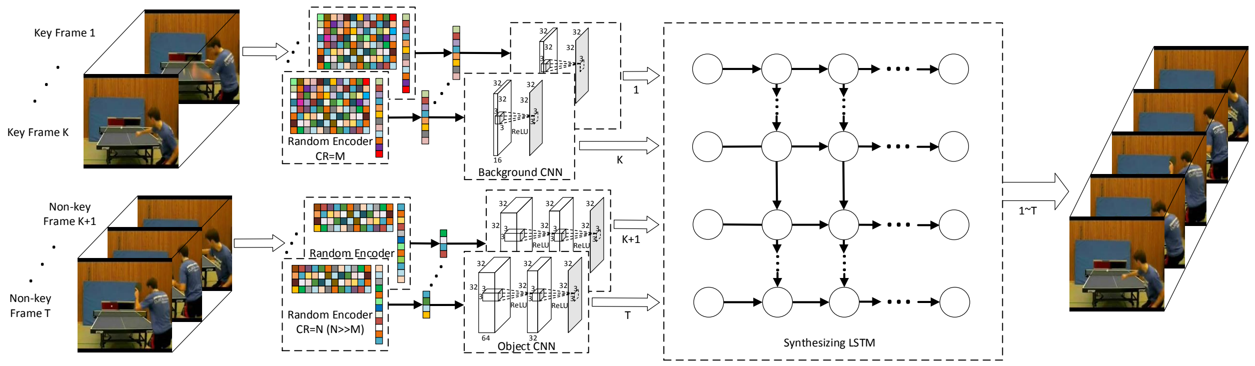


Figure 1. The overall architecture of the proposed framework. The compressed video frames are acquired by compressive sensing. The reconstruction is performed by the CSVideoNet consisting of a background CNN, an object CNN, and a synthesizing LSTM. In every T frames, the first M and the remaining (T-M) frames are compressed with a low and high CR, respectively. The background CNN is first pre-trained. Then, the last a few layers of the background CNN are jointly trained with the rest of the model.

ing LSTM, the three networks are trained together, shown in Figure 4. To reduce the number of parameters needed for training, only the last few layers of key-frame CNNs are incorporated, that's the reason why the input for these layers are features maps rather than measurements.

The CNN in CSVideoNet is pre-trained by Caffe, and the whole framework is trained by Torch. We use average Euclidean loss as the loss function, which is $L(\mathbf{W},\mathbf{b}) = \frac{1}{2N}\sum_i^T \|f(y_i,\mathbf{W},\mathbf{b}) - x_i\|_2^2$, where $\mathbf{W}$ and $\mathbf{b}$ are the network weight and bias, respectively, x_i and y_i are the i-th image blocks and its CS measurement. A Random Gaussian matrix is used for CS encoding.

4. Experiment Result

4.1. Dataset

We evaluate our model on the UCF-101 dataset. The UCF-101 database consists of 13k clips and 27 hours of video data collected from YouTube, which belong to 101 action classes. The videos have a resolution of 320×240 pixels and are sampled at approximately 30 fps. We crop each video frame into a 160×160 patch by only keeping the central pixels as valid data. These patches are then segmented into non-overlapping 32×32 blocks for training. Our model is tested at the average compression ratio of 25, 50 and 100, respectively. Since the size of each input image block is 32×32, the corresponding dimension of measurements is 40, 20 and 10, respectively. We retain only the luminance component of the extracted patches. The dimension of training data for the CNNs is $(N \times C \times H \times W)$, where N=100 is the batch size, C=1 is the channel size, H=32 and W=32 is the height and width of each frame, respectively. The dimension of the training data for LSTM is $(N \times T \times C \times H \times W)$, where T=10 is the sequence length, N=20 is the batch size, and the other dimensions are the same as the CNNs. In every sequence of 10 consecutive video frames, we set the first frame as the key frame, and the rest 9 frames as non-key frames.

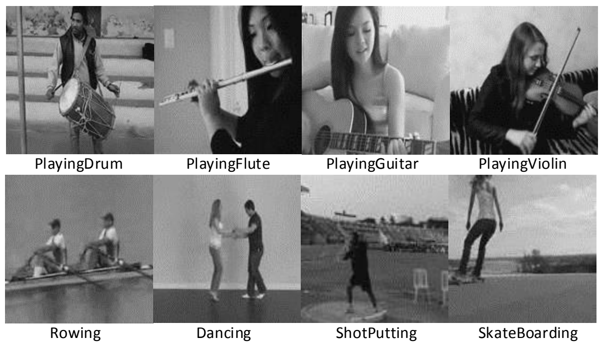


Figure 2. Example of test frames extracted from the video clips.

We extract one frame of each test video and list all the extracted test frames in Figure 2. The test videos contain strenuous sports such as rowing, shot putting, and skateboarding, as well as relatively slow movements, such as playing instruments.

4.2. Performance Under Large-scale Video Dataset

We compare our algorithm with two reference work for video compressive sensing: [39, 26]. We use the codes available from the authors' websites for testing the reference methods. For a fair comparison, we also tune the parameters of the reference algorithms to better fit the UCF-101 dataset.

[39] use a fixed CR for all the video frames. Differently, [26] and our work employ a multi-level and variable CR. The experiments for [39] are performed at the CRs of 25, 50, 100. Differently, the experi-

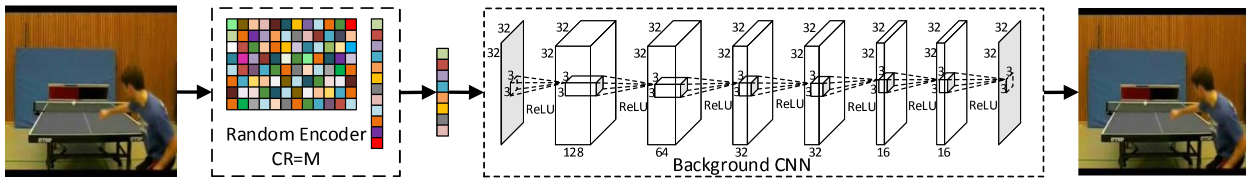


Figure 3. The pre-training of the background CNN.

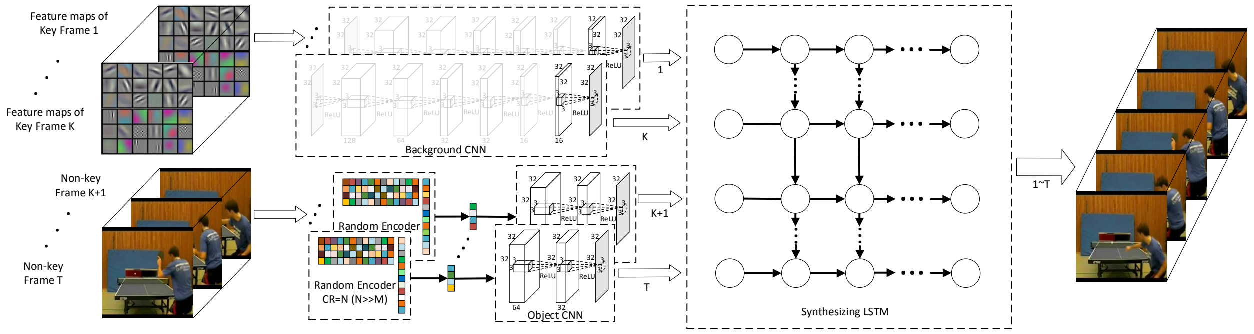


Figure 4. The joint training of the CNNs and the synthesizing LSTM.

ments for [26] and our model are performed at the multi-level CRs of 5/25, 5/50 and 5/100, where the first and second number is the CR of the key and non-key frames, respectively. As 90% of the frames are non-key, the averaged compression ratio is dominated by that of the non-key frames. Peak signal-to-noise ratio (PSNR), structural similarity (SSIM) index, and least absolute errors (LAE) are used as the metrics in the performance evaluation. The experiment results for each method under different CRs are summarized in Table 1. The mean PSNR, SSIM, and LAE across different video types are shown in Figure 6. It is shown that CSVideoNet outperforms the reference methods on all the three metrics, especially at the high CRs. The pixel-wise absolute error of CSVideoNet is approximately 4 for all the CRs. To demonstrate the impact of LSTM on reconstruction performance, we compare the PSNR achieved by CSVideoNet with and without the LSTM network (the CNNs only). The comparison result is shown in Table 2. The reconstruction quality of CSVideoNet is constantly better than that of the CNNs only across different CRs. One should note that the CSVideoNet performance at the CR of 100 is almost comparable to that of the CNNs only at the CR of 25. This validates our hypothesis that LSTM can exploit the temporal correlations between adjacent frames to significantly enhance the CS performance of video applications in terms of the trade-off between CR and reconstruction quality. Note that the CNN models used in CSVideoNet are similar to the model proposed in [21] but with a much larger model capacity to fit the dataset. Therefore, the results in Table 2 imply the comparison with the state-of-the-art. To investigate how well the visual information extracted by the CNNs flows in the LSTM, we conduct another set of experiments by feeding both the key frame and the non-key frames into every stage of the LSTM inputs. It is found that the reconstruction performance is almost unchanged. This indicates that the background information of the key frame can be well propagated across LSTM nodes, which is critical to reconstructing high-quality non-key frames. The reconstructed images are illustrated in Figure 6. Some visual details that are well captured by CSVideoNet but not the reference methods are highlighted for comparison purposes.

To demonstrate the noise resilience performances of CSVideoNet, we conducted the video reconstruction experiments again based on noisy measurements. Note that the same network model as in the previous experiments is used here without re-training it to fit the noisy data. We inject Gaussian white noise to the compressive sensing processing at the signal-noise-ratios (SNRs) of 20dB, 40dB, and 60dB. The average PSNR achieved by each method with respect to the CR is shown in Figure 5. It is shown that CSVideoNet can reliably achieve stable PSNRs at different noise energy levels, while outperforms the reference methods on reconstruction quality by a large margin.

We benchmark the runtime performance of different methods. Due to the limitation of the original codes that do not support GPU acceleration, we run the reference methods on an Intel Xeon E5-2600 CPU with 8 cores in parallel, and we accelerated CSVideoNet using a Nvidia GTX Titan X GPU. The runtimes for fully reconstructing a video frame in size of (160×160) are compared in Table 3. It is shown that CSVideoNet only takes 6 to 8 milliseconds for reconstruction, which is equivalent to a reconstruction throughput of 128-166 fps. This is 3 orders of magnitude faster than the reference methods. Through further optimization, we believe CSVideoNet has the potential to be integrated with CS imagers to benefit high-frame-rate video applications.

5. Conclusion

In this paper, we present a deep neural network model—CSVideoNet that can learn visual representations from random measurements for CS video reconstruction. Two CNN models are used to extract background and motion features from lightly and aggressively compressed measurements, respectively. The LSTM network is used to synthesize high-quality frames based on background and motion features by exploiting the strong temporal coherence between adjacent frames. The exploitation of both spatial and temporal features in video frames is the key to enabling aggressive and flexible compression strategy to achieve better CS performance beyond the limitation of conventional methods. CSVideoNet has the potential to be extended as a general framework to decode information directly from compressively sampled time series, i.e., video detection, recognization, and classification tasks.

Table 3. Runtime for reconstructing a 160×160 frame at different CRs using each method.

Model	CR=25	CR=50	CR=100
CSVideoNet	0.008374	0.006474	0.006043
MC-BCS	7.1681	8.0281	9.0056
GMM	8.8691	10.54	18.34

References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. CoRR, abs/1303.5778, 2015.

[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. CoRR, abs/1409.0473, 2014.

[3] 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.

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

[5] E. J. Candès. The restricted isometry property and its implications for compressed sensing. Comptes Rendus Mathematique, 346(9):589 – 592, 2008.

[6] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Transactions on Information Theory, 52(2):489–509, Feb 2006.

[7] X. Chen and C. Lawrence Zitnick. Mind's eye: A recurrent visual representation for image caption generation. In CVPR, 2015.

[8] K. Cho, B. Van Merriënboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014.

[9] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. IEEE Transactions on Image Processing, 16(8):2080–2095, 2007.

[10] 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.

[11] 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.

[12] 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.

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014.

[14] A. Graves and N. Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In ICML, 2014.

[15] S. Harmeling. Image denoising: Can plain neural networks compete with bm3d? In CVPR, 2012.

[16] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Processing Magazine, 29(6):82–97, 2012.

[17] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 313(5786):504–507, 2006.

[18] M. Iliadis et al. Deep Fully-Connected Networks for Video Compressive Sensing. CoRR, abs/1603.04930, 2016.

[19] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In CVPR, 2014.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, 2012.

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

[22] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In CVPR, 2015.

[23] S. Ma, L. Sigal, and S. Sclaroff. Learning activity progression in lstms for activity detection and early detection. In CVPR, 2016.

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

[25] A. Mousavi et al. A Deep Learning Approach to Structured Signal Recovery. CoRR, abs/1508.04065, 2015.

[26] S. Mun and J. E. Fowler. Motion-compensated compressed-sensing reconstruction for dynamic mri. In ICIP, 2013.

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

[28] H. Noh, P. Hongsuck Seo, and B. Han. Image question answering using convolutional neural network with dynamic parameter prediction. In CVPR, 2016.

[29] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Is object localization for free? - weakly-supervised learning with convolutional neural networks. In CVPR, 2015.

[30] S. Qaisar, R. M. Bilal, W. Iqbal, M. Naureen, and S. Lee. Compressive sensing: From theory to applications, a survey. Journal of Communications and Networks, 15(5):443–456, 2013.

[31] 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.

[32] A. C. Sankaranarayanan, C. Studer, and R. G. Baraniuk. Cs-muvi: Video compressive sensing for spatial-multiplexing cameras. In ICCP, 2012.

[33] S. Sharma, R. Kiros, and R. Salakhutdinov. Action recognition using visual attention. In NIPS workshop on Time Series. 2015.

[34] K. Soomro, A. Roshan Zamir, and M. Shah. UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild. CoRR, abs/1212.0402, 2012.

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

[36] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. Information Theory, IEEE Transactions on, 53(12):4655–4666, 2007.

[37] J. Xie, L. Xu, and E. Chen. Image denoising and inpainting with deep neural networks. In NIPS, 2012.

[38] 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-15, 2015.

[39] J. Yang, X. Yuan, X. Liao, P. Llull, D. J. Brady, G. Sapiro, and L. Carin. Video compressive sensing using gaussian mixture models. IEEE Transactions on Image Processing, 23(11):4863–4878, 2014.

[40] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola. Stacked attention networks for image question answering. In CVPR, 2016.

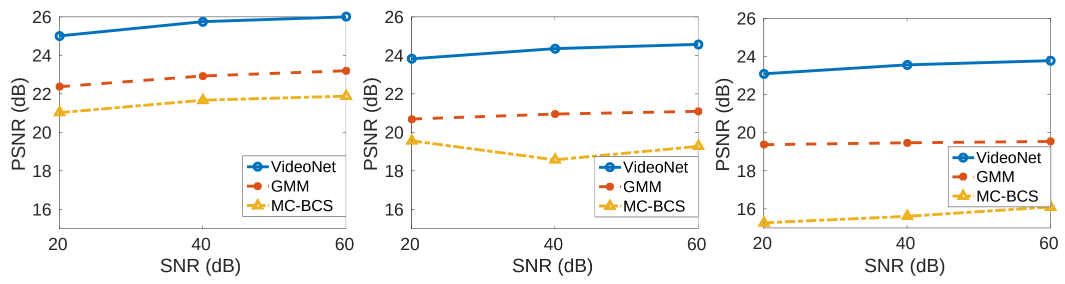
[41] J. Yue-Hei Ng et al. Beyond Short Snippets: Deep Networks for Video Classification. CoRR, abs/1503.08909, 2015.

Table 1. The PSNR, SSIM and LAE performance of each method at the CRs of 25, 50 and 100.

		CR=25			CR=50			CR=100		
Video Name		CSVideoNet	GMM	MC-BCS	CSVideoNet	GMM	MC-BCS	CSVideoNet	GMM	MC-BCS
PlayingDrum	PSNR	24.05	22.67	20.3	22.99	20.2	15.2	22.31	19.47	13.88
	SSIM	0.7	0.65	0.42	0.65	0.5	0.18	0.62	0.44	0.16
	LAE	5.37	6.15	16.45	6.14	9.13	37.16	6.69	10.24	43.89
PlayingFlute	PSNR	28.4	25.88	25.84	26.84	22.72	23.27	25.96	20.64	13.29
	SSIM	0.86	0.89	0.45	0.83	0.69	0.36	0.82	0.65	0.14
	LAE	2.67	2.81	6.33	3.08	4.99	8.61	3.12	7.3	41.25
PlayingGuitar	PSNR	25.87	24.84	21.9	24.55	22.66	20.99	23.78	21.27	18.22
	SSIM	0.79	0.82	0.5	0.75	0.73	0.43	0.73	0.64	0.31
	LAE	3.82	4.06	12.06	4.46	5.73	13.75	4.56	7.31	20.86
PlayingViolin	PSNR	24.18	21.11	19.24	22.32	19.31	18.84	21.43	18.64	13.07
	SSIM	0.74	0.63	0.3	0.67	0.55	0.28	0.64	0.48	0.14
	LAE	4.6	6.89	16.19	5.93	8.85	16.96	6.75	10.19	42.36
Rowing	PSNR	27.43	21.59	23.25	25.58	21.33	21.18	24.75	20.34	22.57
	SSIM	0.83	0.59	0.2	0.77	0.56	0.14	0.74	0.49	0.15
	LAE	2.99	7.21	10.42	3.75	7.56	14.99	4.08	9.1	11.4
Dancing	PSNR	29.56	26.41	23.6	27.51	23.55	22.58	26.38	20.55	22.28
	SSIM	0.91	0.87	0.35	0.87	0.78	0.3	0.85	0.62	0.27
	LAE	1.8	2.88	8.09	2.45	4.73	9.53	2.79	7.56	10.55
ShotPutting	PSNR	29.22	23.63	26.53	27.86	22.05	24.46	27.32	19.81	14.44
	SSIM	0.85	0.7	0.47	0.81	0.63	0.39	0.8	0.55	0.21
	LAE	2.5	5.11	6.7	2.99	6.78	7.97	3.12	8.54	31.85
Skateboarding	PSNR	26.24	23.94	18.64	23.06	18.27	18.24	21.93	16.37	15.64
	SSIM	0.84	0.64	0.29	0.76	0.47	0.29	0.74	0.41	0.18
	LAE	3.26	6.05	18.81	5.66	12.18	19.27	5.6	14.7	28.74
Average	PSNR	26.87	23.76	22.41	25.09	21.26	20.6	24.23	19.64	16.67
	SSIM	0.82	0.72	0.37	0.76	0.61	0.3	0.74	0.54	0.2
	LAE	3.38	5.15	11.88	4.31	7.49	16.03	4.59	9.37	28.86
Improvement w.r.t. Baselines	PSNR	1	13.09%	19.90%	1	18.02%	21.80%	1	23.37%	45.35%
	SSIM	1	13.89%	121.62%	1	24.59%	153.33%	1	37.04%	270.00%
	LAE	1	34.37%	71.55%	1	42.46%	73.11%	1	51.01%	84.10%

Table 2. Mean PSNR achieved by CSVideoNet and CNN-based approaches.

	CR=25		CR=50		CR=100	
Video Name	CSVideoNet	CNN	CSVideoNet	CNN	CSVideoNet	CNN
PlayingDrum	24.05	21.78	22.99	20.21	22.31	18.22
PlayingFlute	28.40	23.82	26.84	21.78	25.96	19.32
PlayingGuitar	25.87	23.14	24.55	21.48	23.78	19.49
PlayingViolin	24.18	22.26	22.32	20.74	21.43	18.99
Rowing	27.43	25.85	25.58	24.17	24.75	22.67
Dancing	29.56	27.02	27.51	25.09	26.38	23.24
ShotPutting	29.22	25.16	27.86	23.59	27.32	21.18
Skateboarding	26.24	25.15	23.06	22.74	21.93	20.43



(a) Mean PSNR at the CR of 25. (b) Mean PSNR at the CR of 50. (c) Mean PSNR at the CR of 100.

Figure 5. Mean PSNR performance of each method when inputs are contaminated by Gaussian white noise with the SNRs of 20dB, 40dB and 60dB.

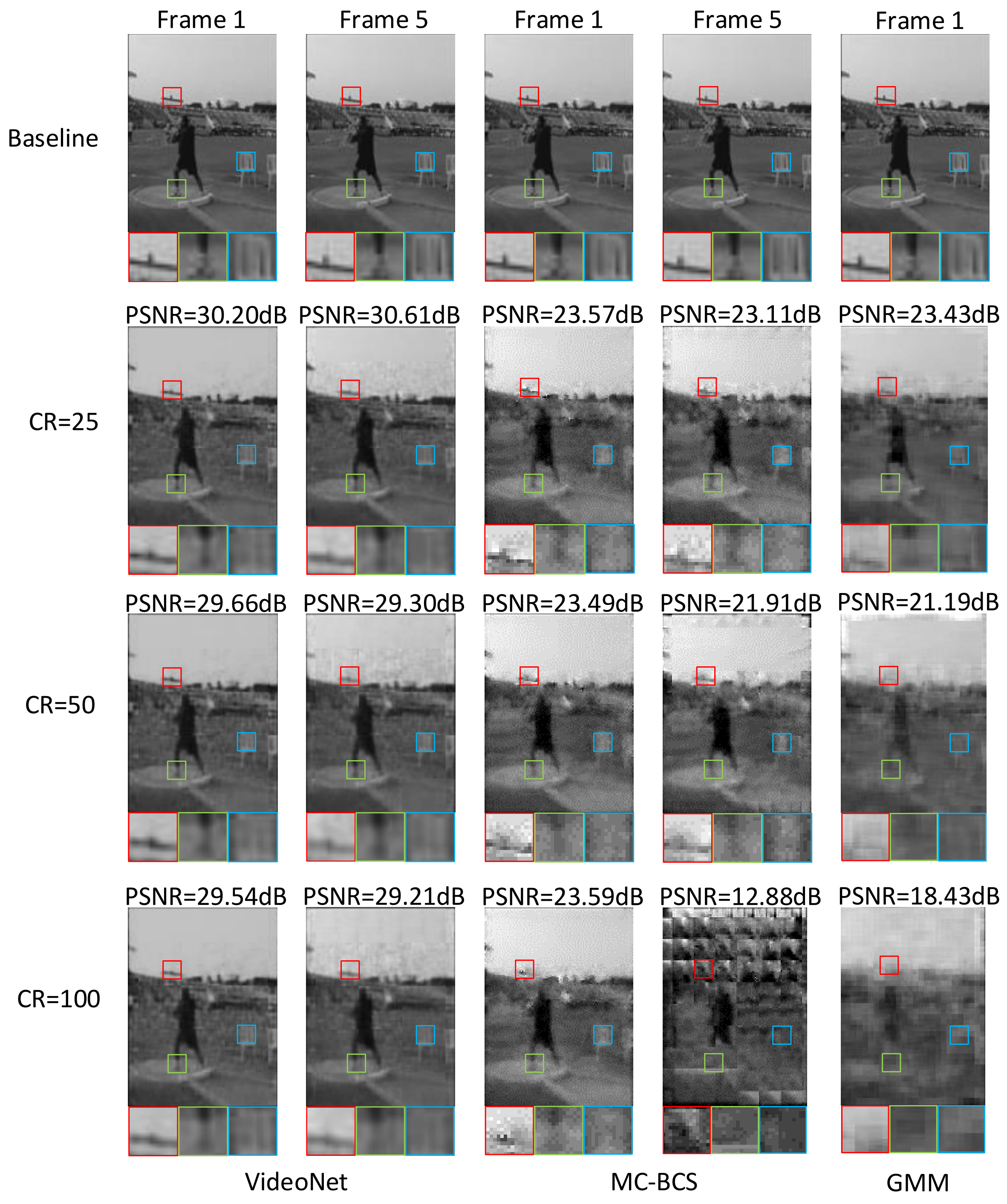


Figure 6. Reconstruction result for each method at the CRs of 25, 50, and 100. Some visual details that are well captured by CSVideoNet but not the reference methods are highlighted for comparison purposes.