Tensor FISTA-Net for Real-Time Snapshot Compressive Imaging

Xiaochen Han,[1] Bo Wu,[2] Zheng Shou,[2] Xiao-Yang Liu,[2*] Yimeng Zhang,[2] Linghe Kong[1]
[1]Shanghai Jiao Tong University, China
[2]Columbia University, USA
{guillermo_han97, linghe.kong}@sjtu.edu.cn, {bo.wu, zs2262, xl2427, yz3397}@columbia.edu

Abstract

Snapshot compressive imaging (SCI) cameras capture high-speed videos by compressing multiple video frames into a measurement frame. However, reconstructing video frames from the compressed measurement frame is challenging. The existing state-of-the-art reconstruction algorithms suffer from low reconstruction quality or heavy time consumption, making them not suitable for real-time applications. In this paper, exploiting the powerful learning ability of deep neural networks (DNN), we propose a novel Tensor Fast Iterative Shrinkage-Thresholding Algorithm Net (*Tensor FISTA-Net*) as a decoder for SCI video cameras. *Tensor FISTA-Net* not only learns the sparsest representation of the video frames through convolution layers, but also reduces the reconstruction time significantly through tensor calculations. Experimental results on synthetic datasets show that the proposed *Tensor FISTA-Net* achieves average PSNR improvement of 1.63∼3.89dB over the state-of-the-art algorithms. Moreover, Tensor FISTA-Net takes less than 2 seconds running time and 12MB memory footprint, making it practical for real-time IoT applications.

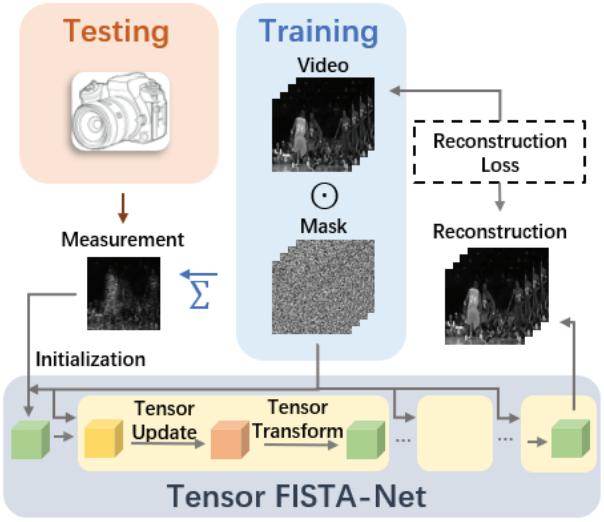

Figure 1: Overview of the *Tensor FISTA-Net*.

Introduction

High-speed cameras are important for sports events, aerial photography and car crashing tests, etc, because slow-motion video recording is needed for several scenarios (Vollmer and Möllmann 2011). Different from conventional high-speed cameras that suffer from limited memory and bandwidth (Saha et al. 2015), the snapshot compressive imaging (SCI) video cameras (Llull et al. 2013), (Gehm et al. 2007), (Wagadarikar et al. 2008) exploit the compressive sensing (CS) theory (Donoho 2006), (Candes, Romberg, and Tao 2006), (Candes and Tao 2006). SCI video cameras adopt sampling on a set of video frames and compress them into a single measurement, which reduces memory and bandwidth cost and enables slow-motion videos and long-time video recording.

Existing algorithms for SCI reconstruction problems are not satisfactory due to exhaustive parameters tuning and heavy time consumption. DeSCI (Liu et al. 2018) is the state-of-the-art algorithm but suffers from extremely long running time due to the non-local patch searching and rank minimization processes. GAP-TV (Yuan 2016) is fast while presents low reconstruction quality due to the low Total Variation (TV) regularizer. Consequently, it is important to design a new method for SCI reconstruction problems with high speed and reconstruction quality.

The FISTA algorithm (Beck and Teboulle 2009) potentially can be used to improve the SCI reconstruction because it can speed up convergence significantly. However, directly using the FISTA algorithm faces the problems of exhaustive parameters tuning and time-consuming large-scale matrix multiplication. Considering the great learning capability of deep neural networks (DNN) and the reduction of calculations in tensor form, in this paper, we propose a novel approach called *Tensor FISTA-Net* for the SCI reconstruction problems. We unfold the FISTA algorithm into a deep neural network inspired by ISTA-Net (Zhang and Ghanem 2018), (Wang, Fidler, and Urtasun 2016), (Frerix et al. 2018), (Jiang et al. 2018) and convert the vector calculations into tensor calculations. Fig. 1 illustrates the overview of the proposed

*Corresponding author.

Tensor FISTA-Net for the SCI video cameras.

The main contributions are summarized as follows:

- We develop propose a novel *Tensor FISTA-Net* for the SCI reconstruction problems by unfolding the FISTA algorithm into a deep neural network.
- We convert the calculations from vector form to tensor form to reduce time and memory consumption significantly. Combining tensors with convolution layers, we explore the sparsest transformation domain of video frames.
- Experimental results on both synthetic and real datasets (collected by SCI video cameras) show that the proposed *Tensor FISTA-Net* outperforms the state-of-the-art algorithms significantly in terms of both reconstruction accuracy and speed. Besides its small model size (12MB), it is practical for real-time applications of IoT devices with limited memory.

The remainder of this paper is organized as follows. We first present the related works for SCI reconstruction problems, then we introduce the mathematical notations and formulate the reconstruction problem of the SCI cameras, after that we propose our *Tensor FISTA-Net* in detail, finally we show and analyze the experimental results.

Related Works

Several optimization-based algorithms have been proposed for SCI reconstruction problems. Sparsity based algorithms (Yuan et al. 2014) has been proposed, (Yang et al. 2014) and (Yang et al. 2015) exploit the sparsity of patches based on Gaussian mixture model (GMM) and sparse coding has been developed in (Wang et al. 2016). Tensor nuclear norm minimization has been adopted in (Liuqing and Liu 2019). DeSCI (Liu et al. 2018) and GAP-TV (Yuan 2016) achieve state-of-the-art performances. However, DeSCI is extremely time-consuming due to the non-local similar patches searching and rank minimization process. DeSCI provides a low reconstruction quality when it is difficult to find non-local patches. GAP-TV is not satisfactory either, it provides reconstruction frames with noise and blur due to the assumption of low total variation.

Neural network-based algorithms include (Iliadis, Spinoulas, and Katsaggelos 2018), (Kai and Ren 2018) and Tensor ADMM-Net (Ma et al. 2019). Tensor ADMM-Net achieves superior performances compared to others. However, the reconstruction results of Tensor ADMM-Net still suffer from noise and blur. This is because Tensor ADMM-Net only learns the linear transformation of video frames through fully connected layers while our *Tensor FISTA-Net* adopts convolution and activation layers to learn more general non-linear transformation. In addition, Tensor ADMM-Net still need vector and matrix form calculations during matrix inversion while calculations in *Tensor FISTA-Net* are all based on tensors.

Reconstruction Problem in SCI Cameras

Overview of the SCI Video Cameras

SCI video cameras have been developed to capture high speed videos in (Hitomi et al. 2011) (Llull et al. 2013) (Yuan et al. 2014). Fig. 2 illustrates the SCI video cameras. The key implementation of SCI cameras is a temporal variant mask. During the measurement process, the mask automatically shifts and multiple video frames are compressed into one measurement frame.

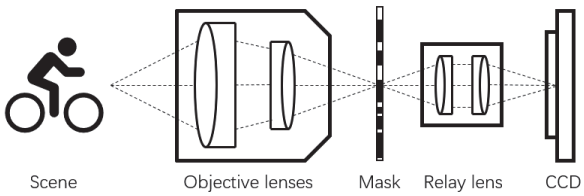

Figure 2: Schema of the SCI video cameras. The mask shifts automatically and the CCD collects the measurement frame.

Consider a B-frame video tensor $\mathcal{X} \in \mathbb{R}^{n\times m\times B}$ and a tensor sensing mask $\mathcal{M} \in \mathbb{R}^{n\times m\times B}$, then the measurement frame $\boldsymbol{Y} \in \mathbb{R}^{n\times m}$ can be expressed as follows (Llull et al. 2013):

$$\boldsymbol{Y} = \sum_{b=1}^{B} \mathcal{M}^{(b)} \odot \mathcal{X}^{(b)}, \tag{1}$$

where $\mathcal{M}^{(b)}$ and $\mathcal{X}^{(b)}$ denote the b-th frontal slice of $\mathcal{M}$ and $\mathcal{X}$, respectively, and $\odot$ denotes the Hadamard (element-wise) product.

Mathematically, (1) is equivalent to the following linear form:

$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x}, \tag{2}$$

where $\boldsymbol{y} = \text{Vec}(\boldsymbol{Y}) \in \mathbb{R}^{nm}$ and $\boldsymbol{x} = [\text{Vec}(\mathcal{X}^{(1)}); ...; \text{Vec}(\mathcal{X}^{(B)})] \in \mathbb{R}^{nmB}$ are the vectorized representation of $\boldsymbol{Y}$ and $\mathcal{X}$, respectively. Different from traditional CS problem, the mask $\boldsymbol{\Phi}$ in (2) is a block diagonal matrix consisting of B diagonal matrices shaped as follows:

$$\boldsymbol{\Phi} = [\text{diag}(\text{Vec}(\mathcal{M}^{(1)})), ..., \text{diag}(\text{Vec}(\mathcal{M}^{(B)}))] \in \mathbb{R}^{n\times mB}. \tag{3}$$

Reconstruction Problem of the SCI Cameras

The reconstruction problem (2) can be solved by the following LASSO (Tibshirani 1996) optimization problem:

$$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{\Psi}\boldsymbol{x}\|_1, \tag{4}$$

where $\boldsymbol{\Psi}\boldsymbol{x}$ denotes the coefficients in transformation domain, $\|\cdot\|_1$ denotes the ℓ_1-norm that imposes the sparsity of the coefficients, and λ balances these two terms.

To solve (4), FISTA algorithm uses the following iterative steps for $k \geq 1$:

$$\boldsymbol{r}^k = \boldsymbol{z}^k - \rho\boldsymbol{\Phi}^T(\boldsymbol{\Phi}\boldsymbol{z}^k - \boldsymbol{y}), \tag{5}$$

$$\boldsymbol{x}^k = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{r}^k\|_2^2 + \lambda\|\boldsymbol{\Psi}\boldsymbol{x}\|_1, \tag{6}$$

$$t^{k+1} = \frac{1 + \sqrt{1 + 4(t^k)^2}}{2}, \tag{7}$$

$$\boldsymbol{z}^{k+1} = \boldsymbol{x}^k + (\frac{t^k - 1}{t^{k+1}})(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}), \tag{8}$$

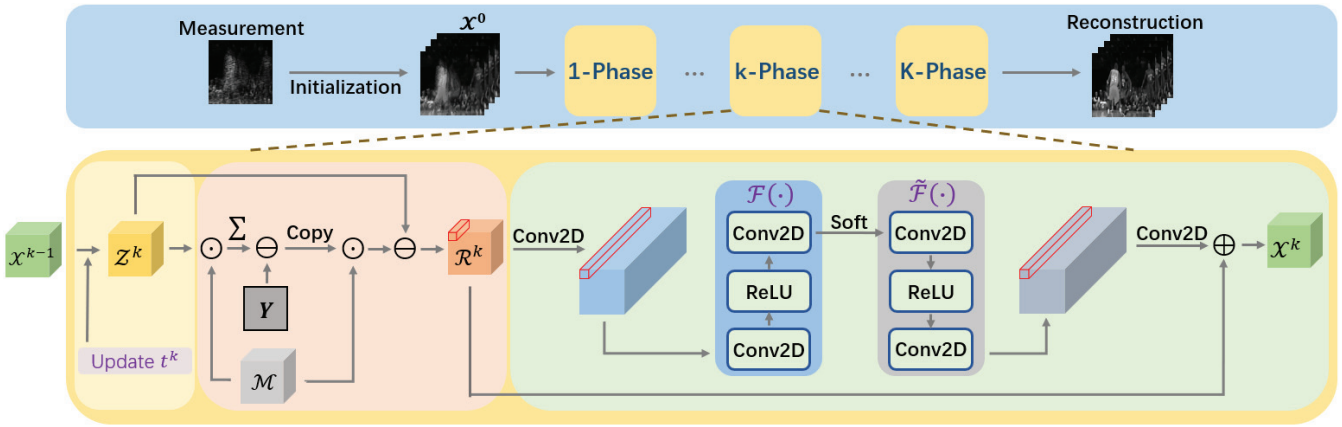

Figure 3: The proposed *Tensor FISTA-Net*. The upper part is the data flow in *Tensor FISTA-Net*, containing K phases. The bottom part is the detailed structure of a phase. Gray arrows denote the data flow.

where $\boldsymbol{z}^1 = \boldsymbol{x}^0$, $t^1 = 1$, ρ represents the step size, $\boldsymbol{r}^k$ is an auxiliary variable, $\boldsymbol{x}^k$ is the output of the k-th iteration and $\boldsymbol{z}^{k+1}$ is a new starting point for next iteration.

The Proposed *Tensor FISTA-Net*

We first describe the framework of *Tensor FISTA-Net* and the motivation for a tensor form neural network, then we elaborate the detailed structure of *Tensor FISTA-Net*.

Tensor FISTA-Net

The basic idea of *Tensor FISTA-Net* is to unfold (5)-(8) into a deep neural network with a fixed number of phases, where each phase corresponds to one iteration. Fig. 3. illustrates the framework of the whole *Tensor FISTA-Net* and the specific design in a phase. Iterations in (5)-(8) are in vector form, instead, we are motivated to design a tensor form neural network due to the following three aspects.

- Temporal correlations within video frames. Unlike a single image, video frames are similar to each other. By stacking them into a tensor in third dimension, we are able to learn the temporal correlations through multi-channel 2D convolution, which better captures the information than that through 2D single-channel convolution in vector form.
- Incorrect features extracted in vector form. In vector form, concatenates one B-frame video into a big image with size $nB \times m$, convolution kernels may extract incorrect spatial information at the borders between two frames. This drawback can be avoided in tensor form.
- Reduction in time and memory consumption through tensor calculations. Due to the special structure of $\boldsymbol{\Phi}$ in (3), updating (5) needs to multiply huge matrices $\boldsymbol{\Phi}^T$ and $\boldsymbol{\Phi}$. However, it will save much time and memory if we update (5) in tensor form (details are shown in **Module** $\mathcal{R}^k$).

Specifically, consider $\mathcal{Z}^k$, $\mathcal{R}^k$ and $\mathcal{X}^k$ as the tensor form of $\boldsymbol{z}^k$, $\boldsymbol{r}^k$ and $\boldsymbol{x}^k$, respectively. Iterative steps in (5)-(8) are considered as four modules: t^k, $\mathcal{Z}^k$, $\mathcal{R}^k$ and $\mathcal{X}^k$ in *Tensor FISTA-Net*. Different from the order in (5)-(8), we calculate t^k and $\mathcal{Z}^k$ before $\mathcal{R}^k$ and $\mathcal{X}^k$ in each phase for simplicity. To elaborate the *Tensor FISTA-Net*, we separately introduce these four modules in the following.

Module t^k : (7) gives an update step of t^{k+1}. However, since deep neural networks have great ability of learning, we set t^k for $k \in [K]$ as learnable variables to improve the flexibility of the proposed *Tensor FISTA-Net*.

Module $\mathcal{Z}^k$: In each phase of *Tensor FISTA-Net*, we directly use the updated t^k to calculate $\mathcal{Z}^k$, thus the tensor form update step of (8) is as follows:

$$\mathcal{Z}^1 = \mathcal{X}^0, \tag{9}$$

$$\mathcal{Z}^k = \mathcal{X}^{k-1} + t^k(\mathcal{X}^{k-1} - \mathcal{X}^{k-2}), \text{ for } k \geq 2. \tag{10}$$

Module $\mathcal{R}^k$: Fig. 4 shows the equivalence transfer of (5) from vector form to tensor form. Consider $\boldsymbol{a} = \boldsymbol{\Phi}\boldsymbol{z}^k - \boldsymbol{y}$ as the middle result of (5), $\boldsymbol{A} \in \mathbb{R}^{n\times m}$ as the matrix form of $\boldsymbol{a}$ and $\text{Vec}(\boldsymbol{A}) = \boldsymbol{a}$, $\mathcal{A} \in \mathbb{R}^{n\times m\times B}$ as the tensor form of $\boldsymbol{a}$ and $\mathcal{A}^{(b)} = \boldsymbol{A}$ for $b \in [B]$. We update $\mathcal{R}^k$ as follows:

$$\mathcal{R}^k = \mathcal{Z}^k - \rho\mathcal{M} \odot \mathcal{A}. \tag{11}$$

For sufficient calculation, (11) can be divided into the following two steps:

(i): Calculate $\boldsymbol{A}$. As illustrated in the first step of Fig 4, $\boldsymbol{A}$ can be calculated as follows combining (1):

$$\boldsymbol{A} = \sum_{b=1}^{B} \mathcal{M}^{(b)} \odot \mathcal{Z}^{k(b)} - \boldsymbol{Y}, \tag{12}$$

where $\mathcal{Z}^k$ is in (9) and (10).

(ii): Calculate $\mathcal{R}^k$, where $\mathcal{R}^k$ is the tensor form of $\boldsymbol{r}^k$ and $\boldsymbol{r}^k = \boldsymbol{z}^k - \rho\boldsymbol{\Phi}^T\boldsymbol{a}$. We know from (3) that $\boldsymbol{\Phi}$ is a block diagonal matrix, thus $\boldsymbol{\Phi}^T \in \mathbb{R}^{nmB\times nm}$ is also a block diagonal matrix. Then $\boldsymbol{\Phi}^T\boldsymbol{a} = [\text{diag}(\text{Vec}(\mathcal{M}^{(1)})) \cdot \boldsymbol{a}; ...; \text{diag}(\text{Vec}(\mathcal{M}^{(B)})) \cdot \boldsymbol{a}] \in \mathbb{R}^{nmB\times nm}$. Specifically, since $\text{diag}(\text{Vec}(\mathcal{M}^{(b)})) \cdot \boldsymbol{a} = \mathcal{M}^{(b)} \odot \boldsymbol{A}$ for $b \in [B]$, in order to calculate $\boldsymbol{\Phi}^T\boldsymbol{a}$ more efficiently in tensor form, we need to copy $\boldsymbol{A} \in \mathbb{R}^{n\times m}$ for B times to fold it into a tensor

$\mathcal{A} \in \mathbb{R}^{n\times m\times B}$ so that $\mathcal{A}^{(b)} = \boldsymbol{A}$ for $b \in [B]$. Then $\mathcal{R}^k$ can be updated in tensor form as (11).

Module $\mathcal{X}^k$: (6) uses a sparse transformation as $\boldsymbol{\Psi}$. However, we aim to find the sparest transformation domain rather than using a pre-set one. Inspired by (Wu et al. 2016b) and (Wu et al. 2016a), we take the temporal correlations into consideration and design the general transform based on tensor form.

Consider the great representation power of convolution neural network (CNN) (Dong et al. 2014) and its universal approximation property (Hornik, Stinchcombe, and White 1989), we use convolution and activation layers to represent the transformation function denoted by $\mathcal{F}(\cdot)$, which is supposed to learn the sparsest representation of video frames. Specifically, $\mathcal{F}(\cdot)$ contains two multi-channel 2D convolution layers split by one ReLU activation layer, we denote it as $\mathcal{F}(\mathcal{X}) = \text{Conv2D}(\text{ReLU}(\text{Conv2D}(\mathcal{X})))$.

Replace $\boldsymbol{\Psi}$ with $\mathcal{F}(\cdot)$, the tensor form of (6) is as follows:

$$\mathcal{X}^k = \arg\min_{\mathcal{X}} \frac{1}{2}\|\mathcal{X} - \mathcal{R}^k\|_F^2 + \lambda\|\mathcal{F}(\mathcal{X})\|_1. \tag{13}$$

Actually, $\mathcal{F}(\boldsymbol{x})$ is equivalent to perform a matrix multiplication to $\boldsymbol{x}$, i.e. $\mathcal{F}(\boldsymbol{x}) = \boldsymbol{D}\boldsymbol{x}$, where $\boldsymbol{D}$ is an orthonormal matrix consisting of all basis vectors in transformation domain.

Theorem 1

$$\|\boldsymbol{D}\boldsymbol{a} - \boldsymbol{D}\boldsymbol{b}\|_2 = \|\boldsymbol{a} - \boldsymbol{b}\|_2, \tag{14}$$

where $\boldsymbol{D}$ is an orthonormal transformation matrix.

Proof. It is a special case of the Parseval Theorem. $\boldsymbol{D}$ is orthonormal with spectrum norm $\|\boldsymbol{D}\| = 1$. Therefore, $\|\boldsymbol{D}\boldsymbol{a} - \boldsymbol{D}\boldsymbol{b}\|_2 = \|\boldsymbol{D}(\boldsymbol{a} - \boldsymbol{b})\|_2 = \|\boldsymbol{D}\| \cdot \|\boldsymbol{a} - \boldsymbol{b}\|_2 = \|\boldsymbol{a} - \boldsymbol{b}\|_2$. □

With Theorem 1, $\|\boldsymbol{x} - \boldsymbol{r}^k\|_2^2 = \|\boldsymbol{D}\boldsymbol{x} - \boldsymbol{D}\boldsymbol{r}^k\|_2^2 = \|\mathcal{F}(\boldsymbol{x}) - \mathcal{F}(\boldsymbol{r}^k)\|_2^2$. Since $\|\cdot\|_2$ denotes the ℓ_2-norm, which is the square root of the sum of the square of all elements, Theorem 1 also holds when it is extended to tensor form, so we replace $\|\mathcal{F}(\boldsymbol{x}) - \mathcal{F}(\boldsymbol{r}^k)\|_2^2$ by $\|\mathcal{F}(\mathcal{X}) - \mathcal{F}(\mathcal{R}^k)\|_F^2$ and (13) becomes :

$$\mathcal{X}^k = \arg\min_{\mathcal{X}} \frac{1}{2}\|\mathcal{F}(\mathcal{X}) - \mathcal{F}(\mathcal{R}^k)\|_F^2 + \lambda\|\mathcal{F}(\mathcal{X})\|_1. \tag{15}$$

Soft-thresholding operator (Donoho 1995) is an element-wise operator when applied to a tensor, we adopt it to obtain a closed-form solution of $\mathcal{F}(\mathcal{X}^k)$:

$$\mathcal{F}(\mathcal{X}^k) = \text{soft}(\mathcal{F}(\mathcal{R}^k), \lambda). \tag{16}$$

To get $\mathcal{X}^k$ from $\mathcal{F}(\mathcal{X}^k)$, we introduce an inverse transformation function $\widetilde{\mathcal{F}}(\cdot)$ to inverse the transformation function $\mathcal{F}(\cdot)$. $\widetilde{\mathcal{F}}(\cdot)$ is supposed to have the ability of inversion, i.e. $\widetilde{\mathcal{F}}(\mathcal{F}(\mathcal{X})) = \mathcal{X}$. Therefore, the inverse function $\widetilde{\mathcal{F}}(\cdot)$ is realized by using the symmetric structure of $\mathcal{F}(\cdot)$, so it can be written as $\widetilde{\mathcal{F}}(\mathcal{X}) = \text{Conv2D}(\text{ReLU}(\text{Conv2D}(\mathcal{X})))$, the only difference between $\mathcal{F}(\cdot)$ and $\widetilde{\mathcal{F}}(\cdot)$ is the convolution kernels.

Then we adopt the inverse transformation function $\widetilde{\mathcal{F}}(\cdot)$ and obtain the closed-form solution of $\mathcal{X}^k$:

$$\mathcal{X}^k = \widetilde{\mathcal{F}}(\text{soft}(\mathcal{F}(\mathcal{R}^k), \lambda)). \tag{17}$$

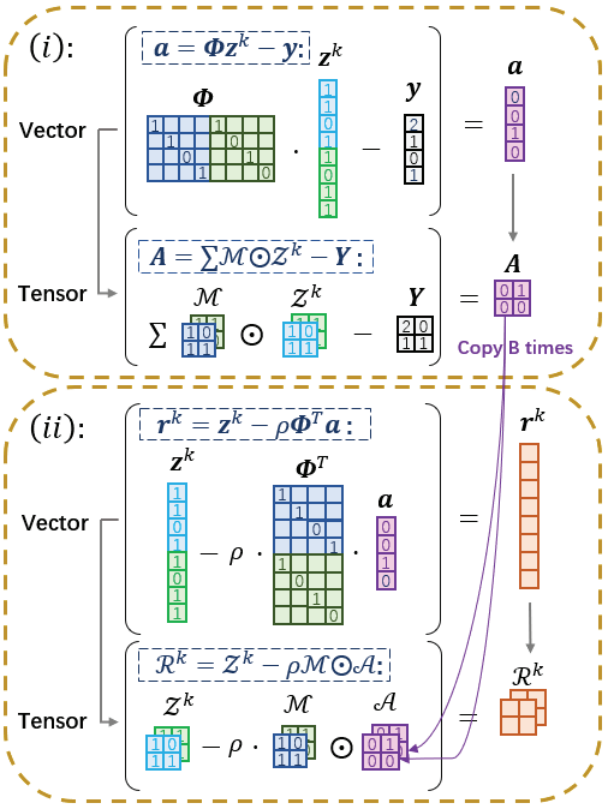

Figure 4: Illustration of tensor calculations of $\mathcal{R}^k$. $\boldsymbol{A}$ in the first step is the middle result of calculation. In the second step $\boldsymbol{A}$ is copied B times to form tensor $\mathcal{A}$ in third dimension.

Finally, inspired by Residual Network (He et al. 2016), we add a shortcut structure and the solution of $\mathcal{X}^k$ is as follows:

$$\mathcal{X}^k = \mathcal{R}^k + \widetilde{\mathcal{F}}(\text{soft}(\mathcal{F}(\mathcal{R}^k), \lambda)). \tag{18}$$

Design of $\mathcal{F}(\cdot)$ and $\widetilde{\mathcal{F}}(\cdot)$

In *Tensor FISTA-Net*, convolution layers perform the linear transformation for video frames, and activation layers introduce nonlinearity to the transformation and impose its sparsity. Specifically, we use two convolution layers split by ReLU layer to serve as the transformation structure, denoted by $\mathcal{F}(\mathcal{X}) = \boldsymbol{C}_2(\boldsymbol{R}(\boldsymbol{C}_1(\mathcal{X})))$, where $\boldsymbol{C}_1$, and $\boldsymbol{C}_2$ denote 2 convolution layers with different convolution kernels, respectively. Similarly, the inverse transformation function $\widetilde{\mathcal{F}}(\cdot)$ can be expressed as $\widetilde{\mathcal{F}}(\mathcal{X}) = \widetilde{\boldsymbol{C}}_2(\boldsymbol{R}(\widetilde{\boldsymbol{C}}_1(\mathcal{X})))$, where $\widetilde{\boldsymbol{C}}_1$ and $\widetilde{\boldsymbol{C}}_2$ denote two convolution layers with different convolution kernels, respectively.

Consider the close temporal correlations among the frontal slices of the video tensors, we use multi-channel 2D convolution because it extracts information among temporal sequence and learns sprase representation of the video tensors. For simplicity, we use the same kernel size ($3 \times 3 \times B$) in all convolution layers.

Additionally, in order to improve the representation ability, we use an extra convolution layer to increase the number of channels from B to 64 before the transformation function $\mathcal{F}(\cdot)$ and use another convolution layer to reduce the number of channels from 64 to B after the inverse transformation function $\widetilde{\mathcal{F}}(\cdot)$. Specifically, we keep the input, output channels as 64 in $\mathcal{F}(\cdot)$ and $\widetilde{\mathcal{F}}(\cdot)$.

Initialization

A proper initialization helps reduce training time significantly. Naturally, consider the special structure of the mask $\mathbf{\Phi}$ in (2), we initialize each frontal slice of $\mathcal{X}^0$ as the measurement frame divided by the sum of mask in third dimension. Consider $\boldsymbol{M} \in \mathbb{R}^{n\times m}$ to be the sum of mask in third dimension, $\mathcal{X}^0 \in \mathbb{R}^{n\times m\times B}$ can be initialized as follows:

$$\boldsymbol{M} = \sum_{b=1}^{B} \mathcal{M}^{(b)}, \tag{19}$$

$$\mathcal{X}^{0(b)} = \boldsymbol{Y} \oslash \boldsymbol{M}, \text{ for } b \in [B], \tag{20}$$

where $\oslash$ denotes element-wise divide between two matrices.

Loss Function

Three constraints should be taken into consideration in *Tensor FISTA-Net*:

- The fidelity of the reconstructed frames.
- The Accuracy of the inverse function.
- The sparsity of video frames in the learned domain.

Assume there are K phases in total, $\mathcal{X}^k \in \mathbb{R}^{n\times m\times B}$ for $k \in [K]$ is the output of the k-th phase, the three constraints can be expressed as follows:

$$\mathcal{L}_{\text{fidelity}} = \|\mathcal{X}^K - \mathcal{X}\|_F^2, \tag{21}$$

$$\mathcal{L}_{\text{inversion}} = \frac{1}{K}\sum_{k=1}^{K} \|\widetilde{\mathcal{F}}(\mathcal{F}(\mathcal{X}^k)) - \mathcal{X}^k\|_F^2, \tag{22}$$

$$\mathcal{L}_{\text{sparsity}} = \frac{1}{K}\sum_{k=1}^{K} \|\mathcal{F}(\mathcal{X}^k)\|_1. \tag{23}$$

The loss function is a weighted sum of these three terms:

$$\mathcal{L} = \alpha\mathcal{L}_{\text{fidelity}} + \beta\mathcal{L}_{\text{inversion}} + \gamma\mathcal{L}_{\text{sparsity}}, \tag{24}$$

where α, β and γ are coefficients to balance the 3 terms. In default, we set $\alpha = 1$, $\beta = 0.01$ and $\gamma = 0.001$.

Performance Evaluation

Datasets

We evaluate the proposed *Tensor FISTA-Net* on three different synthetic datasets: `Kobe` (Yang et al. 2014), `Park` (Ma et al. 2019) and `Vehicle` (Ma et al. 2019). Each testing dataset contains 32 frames of size 256×256 and $B = 8$, i.e., 4 measurements. We use the same training videos NBA, Central Park Aerial and Vehicle Crashing Tests in (Ma et al. 2019). We resize the video frames into 256×256 through down sampling and extract the luminance component to make the training datasets, each training dataset contains 8000 frames, i.e., 1000 measurements.

To compare with previous algorithms, we set $B = 8$ and use the same synthetic mask following the settings in (Liu et al. 2018). Peak Signal to Noise Ratio (PSNR), Structural Similarity Index (SSIM) (Zhou et al. 2004) and Reconstruction Time are performance metrics in comparison.

Table 1: Average PSNR (dB) on different datasets.

Algorithm	Kobe	Park	Vehicle
Tensor FISTA-Net	31.41	**27.64**	**26.46**
GAP-TV	26.45	24.53	22.85
DeSCI	**33.25**	24.95	21.16
Tensor ADMM-Net	30.15	26.85	23.62

Table 2: Average SSIM on different datasets.

Algorithm	Kobe	Park	Vehicle
Tensor FISTA-Net	0.92	**0.88**	**0.89**
GAP-TV	0.84	0.84	0.77
DeSCI	**0.95**	0.80	0.70
Tensor ADMM-Net	0.89	0.86	0.78

Table 3: Running time (seconds) on different datasets.

Algorithm	Kobe	Park	Vehicle
Tensor FISTA-Net	**1.5**	**1.8**	**1.8**
GAP-TV	7.9	6.9	7.2
DeSCI	6872.9	6915.8	6823.5
Tensor ADMM-Net	1.9	2.4	2.1

Implementation details

We use TensorFlow to implement our algorithm and do all the experiments on a server with an NVIDIA Telsa V100-PCIE GPU (16GB device memory). We set the number of phases as 10, learning rate as 0.0001 and running epoch as 500. Adam optimizer (Kingma and Ba 2014) is used to minimize the training loss. The code is available at https://github.com/GuillermoHan97/SCI-Tensor-FISTA-Net.

Comparison with State-of-the-Art Algorithms

GAP-TV (Yuan 2016) adopts the idea of total variance minimization under the generalized alternating projection (GAP). It is simple and fast, thus can be used as a baseline for the experimental results.

DeSCI (Liu et al. 2018) exploits rank minimization for non-local patches and achieve the best results among optimization-based algorithms. **Tensor ADMM-Net** (Ma et al. 2019) is the state-of-the-art network-based algorithms unfolding the ADMM algorithm into a DNN.

Performance Comparison. To compare the reconstruction accuracy of different algorithms, we calculate PSNR

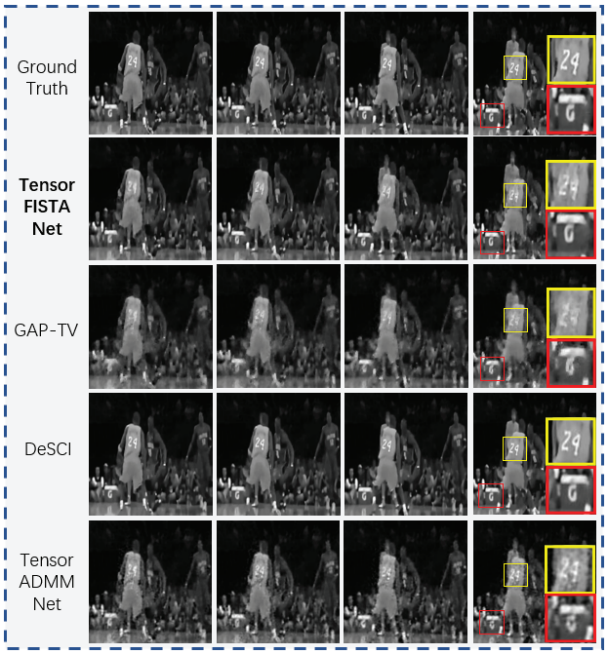

Figure 5: On Kobe dataset (256×256, $B = 8$): Four selected Ground Truth and reconstruction frames.

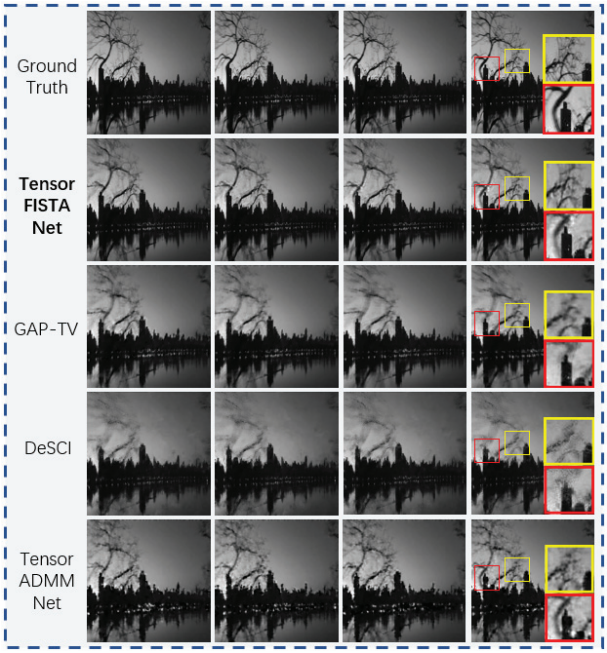

Figure 6: On Park dataset (256×256, $B = 8$): Four selected Ground Truth and reconstruction frames.

and SSIM of the experimental results. Tables. 1-2 show the reconstruction accuracy of three synthetic datasets using *Tensor FISTA-Net* and other algorithms. On Kobe dataset, DeSCI provides the best results. Our *Tensor FISTA-Net* provides the best results on Park dataset (0.79dB in PSNR and 0.02 in SSIM higher than the state-of-the-art algorithm) and

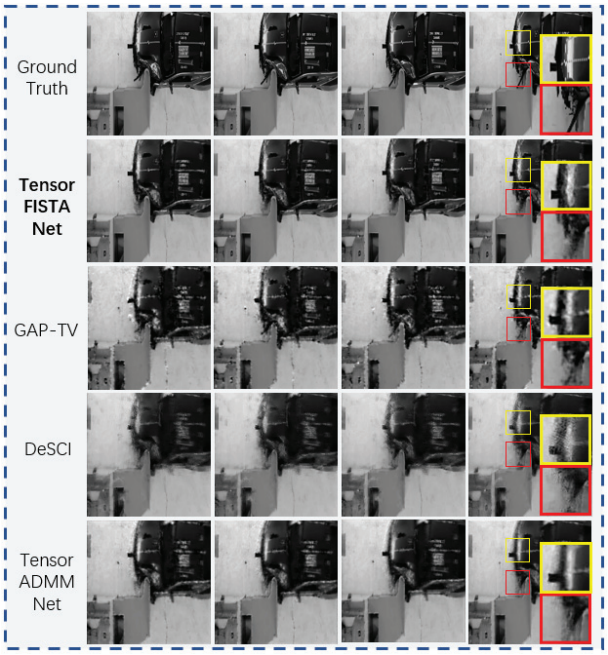

Figure 7: On Vehicle dataset (256×256, $B = 8$): Four selected Ground Truth and reconstruction frames.

Vehicle dataset (2.84dB in PSNR and 0.11 in SSIM higher than the state-of-the-art algorithm).

To validate the reconstruction results, we show the reconstruction frames of three datasets using different algorithms in Figs. 5-7. Since the training and testing datasets of Kobe are not splited from the same video, our *Tensor FISTA-Net* do not achieve the best, but it still provide better results than other algorithms except DeSCI. From the reconstruction frames of Park and Vehicle datasets, we observe that DeSCI can not reconstruct the details of the videos, especially when a small area contains many details like the branches in Park dataset and edges of cars in Vehicle dataset (marked by yellow square). Moreover, we notice that DeSCI suffers from over-smooth and it smooths out tiny details. This indicates that if similar patches cannot be found in video frames, the reconstruction quality of DeSCI will be limited. Tensor ADMM-Net provides better results than DeSCI in Park and Vehicle datasets, but it still suffers from blur and noise compared with our *Tensor FISTA-Net* such as tower in Park dataset (marked by red square) and edges of cars in Vehicle dataset (marked by yellow square).

Time Complexity Analysis. For the time complexity analysis of different algorithms, we record the running time on three synthetic datasets of different algorithms. Table. 3 shows that the reconstruction speed of *Tensor FISTA-Net* is over 4 times faster than GAP-TV and even $4K+$ times faster than DeSCI in average. In fact, *Tensor FISTA-Net* runs in less than 2 seconds to reconstruct one measurement, which is even faster than Tensor ADMM-Net, too.

In addition, the size of our neural network model is less than 12MB, so it is suitable for IoT devices with small memory such as drones.

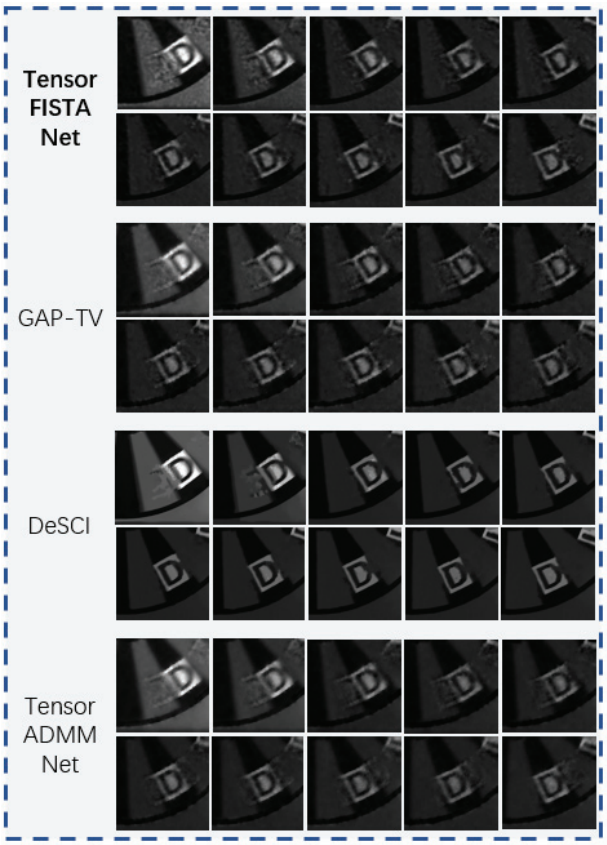

Figure 8: On `Wheel` dataset (256×256, $B = 14$): Ten selected real gray SCI measurement reconstruction frames.

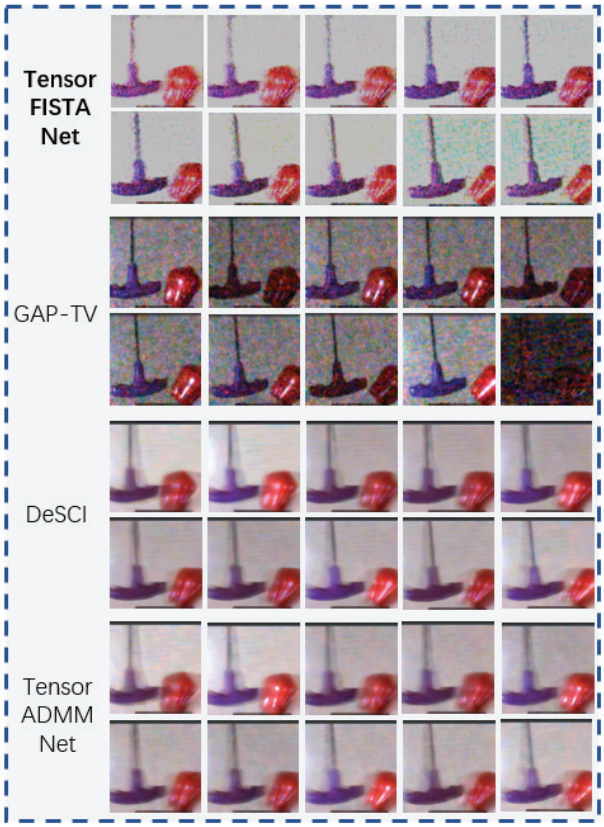

Figure 9: On `Hammer` dataset (512×512, $B = 22$): Ten selected real bayer RGB SCI measurement reconstruction frames.

Real SCI Datasets

We apply the proposed *Tensor FISTA-Net* on real SCI datasets `Wheel` (gray scale, $B = 14$) and `Hammer` (bayer RGB, $B = 22$) captured by SCI cameras (Llull et al. 2013). For colored datasets, the reconstruction can be done by separately reconstructing RGB channels and then aggregating them. These two real datasets compress more video frames into one measurement than synthetic datasets, which is a greater challenge for us. Fig. 8 and Fig. 9 show ten selected reconstruction frames of `Wheel` and `Hammer` datasets, respectively. From the reconstruction frames we observe that the proposed *Tensor FISTA-Net* provides clear edges and less noise in `Wheel` dataset. In `Hammer` dataset, *Tensor FISTA-Net* provides the reconstruction frames with less phantom than DeSCI and Tensor ADMM-Net and much less noise than GAP-TV.

Conclusion

In this paper, we proposed *Tensor FISTA-Net* by unfolding the FISTA algorithm into a deep neural network for the SCI reconstruction problem and transferring the calculations from vector form to tensor form. Experimental results show that *Tensor FISTA-Net* provides better reconstruction quality and runs faster than the state-of-the-art algorithms. In addition, *Tensor FISTA-Net* consumes much less memory thus is suitable for real-time applications on IoT devices.

Acknowledgement

Linghe Kong is supported by NSFC 61972253, 61672349, U190820096. This work is done during Xiaochen Han's summer research intern in Columbia University.

References

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

Candes, E. J., and Tao, T. 2006. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory* 52(12):5406–5425.

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

Dong, C.; Loy, C. C.; He, K.; and Tang, X. 2014. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*, pp. 184–199.

Donoho, D. L. 1995. De-noising by soft-thresholding. *IEEE Transactions on Information Theory* 41(3):613–627.

Donoho, D. L. 2006. Compressed sensing. *IEEE Transactions on Information Theory* 52(4):1289–1306.

Frerix, T.; Möllenhoff, T.; Moeller, M.; and Cremers, D. 2018. Proximal backpropagation. In *International Conference on Learning Representations*.

Gehm, M.; John, R.; Brady, D.; Willett, R.; and Schulz, T. 2007. Single-shot compressive spectral imaging with a dual-disperser architecture. *Optics Express* 15(21):14013–14027.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778.

Hitomi, Y.; Gu, J.; Gupta, M.; Mitsunaga, T.; and Nayar, S. K. 2011. Video from a single coded exposure photograph using a learned over-complete dictionary. In *International Conference on Computer Vision*, pp. 287–294.

Hornik, K.; Stinchcombe, M.; and White, H. 1989. Multilayer feedforward networks are universal approximators. *Neural networks* 2(5):359–366.

Iliadis, M.; Spinoulas, L.; and Katsaggelos, A. K. 2018. Deep fully-connected networks for video compressive sensing. *Digital Signal Processing* 72:9–18.

Jiang, F.; Liu, X.-Y.; Lu, H.; and Shen, R. 2018. Efficient multi-dimensional tensor sparse coding using t-linear combination. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Kai, X., and Ren, F. 2018. Csvideonet: A real-time end-to-end learning framework for high-frame-rate video compressive sensing. *IEEE Winter Conference on Applications of Computer Vision* pp. 1680–1688.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Liu, Y.; Yuan, X.; Suo, J.; Brady, D.; and Dai, Q. 2018. Rank minimization for snapshot compressive imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence* pp. 1–1.

Liuqing, Y., and Liu, X.-Y. 2019. Tensor nuclear-norm minimization for snapshot compressive imaging cameras. In *MIT Undergraduate Research Technology Conference,*.

Llull, P.; Liao, X.; Yuan, X.; Yang, J.; Kittle, D.; Carin, L.; Sapiro, G.; and Brady, D. J. 2013. Coded aperture compressive temporal imaging. *Opt. Express* 21(9):10526–10545.

Ma, J.; Liu, X.-Y.; Shou, Z.; and Yuan, X. 2019. Deep Tensor ADMM-Net for Snapshot Compressive Imaging. In *The IEEE International Conference on Computer Vision*.

Saha, N.; Ifthekhar, M. S.; Le, N. T.; and Jang, Y. M. 2015. Survey on optical camera communications: challenges and opportunities. *Iet Optoelectronics* 9(5):172–183.

Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 58(1):267–288.

Vollmer, M., and Möllmann, K.-P. 2011. High speed and slow motion: the technology of modern high speed cameras. *Physics Education* 46(2):191.

Wagadarikar, A.; John, R.; Willett, R.; and Brady, D. 2008. Single disperser design for coded aperture snapshot spectral imaging. *Applied Optics* 47(10):B44–B51.

Wang, L.; Xiong, Z.; Shi, G.; Wu, F.; and Zeng, W. 2016. Adaptive nonlocal sparse representation for dual-camera compressive hyperspectral imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39(10):2104–2111.

Wang, S.; Fidler, S.; and Urtasun, R. 2016. Proximal deep structured models. In Lee, D. D.; Sugiyama, M.; Luxburg, U. V.; Guyon, I.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc. 865–873.

Wu, B.; Cheng, W.-H.; Zhang, Y.; and Mei, T. 2016a. Time matters: Multi-scale temporalization of social media popularity. In *Proceedings of the 24th ACM international conference on Multimedia*, 1336–1344.

Wu, B.; Mei, T.; Cheng, W.-H.; and Zhang, Y. 2016b. Unfolding temporal dynamics: Predicting social media popularity using multi-scale temporal decomposition. In *Thirtieth AAAI Conference on Artificial Intelligence*.

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

Yang, J.; Liao, X.; Yuan, X.; Llull, P.; Brady, D. J.; Sapiro, G.; and Carin, L. 2015. Compressive sensing by learning a gaussian mixture model from measurements. *IEEE Transactions on Image Processing* 24(1):106–119.

Yuan, X.; Llull, P.; Liao, X.; Yang, J.; Brady, D. J.; Sapiro, G.; and Carin, L. 2014. Low-cost compressive sensing for color video and depth. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3318–3325.

Yuan, X. 2016. Generalized alternating projection based total variation minimization for compressive sensing. In *IEEE International Conference on Image Processing*, pp. 2539–2543.

Zhang, J., and Ghanem, B. 2018. ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1828–1837.

Zhou, W.; Bovik, A. C.; Sheikh, H. R.; and Simoncelli, E. P. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* 13(4):600–612.