Variational Bayesian Inference for Robust Streaming Tensor Factorization and Completion

Cole Hawkins
Department of Mathematics, UC Santa Barbara
colehawkins@math.ucsb.edu

Zheng Zhang
Department of ECE, UC Santa Barbara
zhengzhang@ece.ucsb.edu

***Abstract*—Streaming tensor factorization is a powerful tool for processing high-volume and multi-way temporal data in Internet networks, recommender systems and image/video data analysis. Existing streaming tensor factorization algorithms rely on least-squares data fitting and they do not possess a mechanism for tensor rank determination. This leaves them susceptible to outliers and vulnerable to over-fitting. This paper presents a Bayesian robust streaming tensor factorization model to identify sparse outliers, automatically determine the underlying tensor rank and accurately fit low-rank structure. We implement our model in Matlab and compare it with existing algorithms on tensor datasets generated from dynamic MRI and Internet traffic.**

I. Introduction

Multi-way data arrays (i.e., tensors) are collected in various application domains including recommender systems [1], computer vision [2], and uncertainty quantification [3]. How to process, analyze and utilize such high-volume tensor data is a fundamental problem in machine learning and signal processing [4]. Effective numerical techniques, such as CANDECOMP/PARAFAC (CP) [5], [6] factorizations, have been proposed to compress full tensors and to obtain low-rank representations. Extensive optimization and statistical techniques have been developed to obtain low-rank factors and to predict the full tensor from an incomplete (and possibly noisy) multi-way data array [7]–[9].

Streaming tensors appear sequentially in the time domain. Incorporating temporal relationships in tensor data analysis can give significant advantages in anomaly detection [10], discussion tracking [11] and context-aware recommender systems [12]. In the past decade, streaming tensor factorization has been studied under several low-rank tensor models, such as the Tucker model in [13] and the CP decomposition in [14]–[17]. Most approaches use similar objective functions, but differ in choosing specific numerical optimization solvers. All existing streaming tensor factorizations assume a fixed rank, and no existing techniques can capture the sparse outliers in a streaming tensor. Several low-rank plus sparse techniques have been proposed for non-streaming tensor data [18]–[20].

Paper Contributions. This paper proposes a method for the *robust factorization and completion* of streaming tensors. We model the whole temporal tensor as the sum of a low-rank streaming tensor and a time-varying sparse component. In order to capture these two different components, we present a Bayesian statistical model to enforce low rank and sparsity. The posterior probability density function (PDF) of the hidden factors is then computed by the variational Bayesian method [21]. Our work can can be regarded as an extension of [19], [20], [22] to streaming tensors with sparse outliers.

II. Preliminaries and Notations

We use a bold lowercase letter (e.g., $\mathbf{a}$) to represent a vector, a bold uppercase letter (e.g., $\mathbf{A}$) to represent a matrix, and a bold calligraphic letter (e.g., $\boldsymbol{\mathcal{A}}$) to represent a tensor. An order-N tensor is an N-way data array $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, where I_k is the size of mode k. Given integer $i_k \in [1, I_k]$ for $k = 1, \ldots, N$, an entry of the tensor $\boldsymbol{\mathcal{A}}$ is denoted as $a_{i_1,\cdots,i_N}$.

Definition II.1. Let $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$ be two tensors of the same dimensions. Their **inner product** is defined as

$$\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle = \sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} a_{i_1,\ldots,i_N} b_{i_1,\ldots,i_N}.$$

The **Frobenius norm** of tensor $\boldsymbol{\mathcal{A}}$ is further defined as

$$||\boldsymbol{\mathcal{A}}||_{\mathrm{F}} = \langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{A}} \rangle^{1/2}. \tag{1}$$

Definition II.2. An N-way tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is **rank-1** if it can be written as a single outer product of N vectors

$$\boldsymbol{\mathcal{T}} = \mathbf{a}^{(1)} \circ \cdots \circ \mathbf{a}^{(N)}, \text{ with } \mathbf{a}^{(k)} \in \mathbb{R}^{I_k} \text{ for } k = 1, \cdots, N.$$

Definition II.3. The **CP factorization** [5], [6] expresses an N-way tensor $\boldsymbol{\mathcal{A}}$ as the sum of multiple rank-1 tensors:

$$\boldsymbol{\mathcal{A}} = \sum_{r=1}^{R} s_r \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(N)}, \text{ with } \mathbf{a}_r^{(k)} \in \mathbb{R}^{I_k}. \tag{2}$$

The minimal integer R that ensures the equality is called the **CP rank** of $\boldsymbol{\mathcal{A}}$. We can also express the CP factorization as

$$\boldsymbol{\mathcal{A}} = \sum_{r=1}^{R} s_r \mathbf{a}_r^{(1)} \circ \cdots \circ \mathbf{a}_r^{(N)} = [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}; \mathbf{s}]\!],$$

where the mode-k factors form the columns of matrix $\mathbf{A}^{(k)}$.

It is convenient to express $\mathbf{A}^{(k)}$ both column-wise and row-wise, so we include two means of expressing a factor matrix

$$\mathbf{A}^{(k)} = [\mathbf{a}_1^{(k)}, \ldots, \mathbf{a}_R^{(k)}] = [\hat{\mathbf{a}}_1^{(k)}; \ldots; \hat{\mathbf{a}}_{I_k}^{(k)}] \in \mathbb{R}^{I_k \times R}.$$

Here $\mathbf{a}_j^{(k)}$ and $\hat{\mathbf{a}}_{i_k}^{(k)}$ denote the j-th column and i_k-th row of $\mathbf{A}^{(k)}$, respectively.

Definition II.4. The **generalized inner product** of N vectors of the same dimension I is defined as

$$\langle \mathbf{a}^{(1)}, \ldots, \mathbf{a}^{(N)} \rangle = \sum_{i=1}^{I} \prod_{k=1}^{N} a_i^{(k)}.$$

With a generalized inner product, the entries of a low-rank tensor $\boldsymbol{\mathcal{A}}$ as in Definition II.3 can be written as:

$$a_{i_1,\ldots,i_N} = \langle \hat{\mathbf{a}}_{i_1}^{(1)}, \ldots, \hat{\mathbf{a}}_{i_N}^{(N)} \rangle.$$

The **Khatri-Rao product** of two matrices $\mathbf{A} \in \mathbb{R}^{I\times R}$ and $\mathbf{B} \in \mathbb{R}^{J\times R}$ is the columnwise Kronecker product:

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1, \ldots, \mathbf{a}_R \otimes \mathbf{b}_R] \in \mathbb{R}^{IJ\times R}.$$

We will use the product notation to denote the Khatri-Rao product of N matrices in reverse order:

$$\bigodot_{n} \mathbf{A}^{(n)} = \mathbf{A}^{(N)} \odot \mathbf{A}^{(N-1)} \odot \cdots \odot \mathbf{A}^{(1)}.$$

If we exclude the k-th factor matrix, the Khatri-Rao product can be written as

$$\bigodot_{n\neq k} \mathbf{A}^{(n)} = \mathbf{A}^{(N)} \odot \cdots \odot \mathbf{A}^{(k+1)} \odot \mathbf{A}^{(k-1)} \odot \mathbf{A}^{(1)}.$$

III. Review of Streaming Tensor Factorization

Let $\{\boldsymbol{\mathcal{X}}_t\}$ be a temporal sequence of N-way tensors, where $t \in \mathbb{N}$ is the time index and the tensor $\boldsymbol{\mathcal{X}}_t$ of size $I_1 \times \cdots \times I_N$ is a slice of this multi-way stream. Streaming tensor factorization aims to extract the latent CP factors evolving with time.

The standard streaming tensor factorization is [16]:

$$\min_{\{\mathbf{A}^{(k)}\}_{k=1}^{N+1}} \sum_{t=i}^{T} \mu^{T-t} \|\boldsymbol{\mathcal{X}}_t - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}; \hat{\mathbf{a}}_{t-i+1}^{(N+1)}]\!]\|_F^2. \quad (3)$$

The parameter $\mu \in (0, 1)$ is a forgetting factor that controls the weight of the past data; $\{\mathbf{A}^{(i)}\}$ are the discovered CP factors. Please note that $\hat{\mathbf{a}}_{t-i+1}^{(N+1)}$ denotes one row of the temporal factor matrix $\mathbf{A}^{(N+1)}$. The sliding window size $T - i + 1$ can be specified by the user.

In many applications, only partial data $\boldsymbol{\mathcal{X}}_{t,\Omega_t}$ is observed at each time point. Here Ω_t denotes the index set of the partially observed entries. For a general N-way tensor $\boldsymbol{\mathcal{X}}$ and a sampling set Ω, we have

$$\boldsymbol{\mathcal{X}}_\Omega = \begin{cases} x_{i_1,\cdots,i_N} & \text{if } (i_1, i_2, \cdots, i_N) \in \Omega \\ 0 & \text{otherwise.} \end{cases}$$

In such cases, the underlying hidden factors can be computed by solving the following *streaming tensor completion* problem:

$$\min_{\{\mathbf{A}^{(k)}\}_{k=1}^{N+1}} \sum_{t=i}^{T} \mu^{T-t} \| \left(\boldsymbol{\mathcal{X}}_t - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}; \hat{\mathbf{a}}_{t-i+1}^{(N+1)}]\!] \right)_{\Omega_t} \|_{\mathrm{F}}^2. \quad (4)$$

Existing streaming factorization and completion frameworks [15]–[17] solve (3) and (4) as follows: at each time step one updates the N non-temporal factor matrices $\mathbf{A}^{(j)} \in \mathbb{R}^{I_j\times R}$ and $\{\hat{\mathbf{a}}_{t-i+1}^{(N+1)}\}$. By fixing the past time factors, these approaches provide an efficient updating scheme to solve the above non-convex problems.

IV. Bayesian Model for Robust Streaming Tensor Factorization & Completion

In this section, we present a Bayesian method for the robust factorization and completion of streaming tensors $\{\boldsymbol{\mathcal{X}}_t\}$.

A. An Optimization Perspective

In order to simultaneously capture the sparse outliers and the underlying low-rank structure of a streaming tensor, we assume that each tensor slice $\boldsymbol{\mathcal{X}}_t$ can be fit by

$$\boldsymbol{\mathcal{X}}_t = \tilde{\boldsymbol{\mathcal{X}}}_t + \boldsymbol{\mathcal{S}}_t + \boldsymbol{\mathcal{E}}_t. \quad (5)$$

Here $\tilde{\boldsymbol{\mathcal{X}}}_t$ is low-rank, $\boldsymbol{\mathcal{S}}_t$ contains sparse outliers, and $\boldsymbol{\mathcal{E}}_t$ denotes dense noise with small magnitudes. Assume that each slice $\boldsymbol{\mathcal{X}}_t$ is partially observed according to a sampling index set Ω_t. Based on the partial observations $\{\boldsymbol{\mathcal{X}}_{t,\Omega_t}\}$, we will find reasonable low-rank factors for $\{\tilde{\boldsymbol{\mathcal{X}}}_t\}$ in the specified time window $t \in [T - i + 1, T]$ as well as the sparse component $\boldsymbol{\mathcal{S}}_t$. This problem simplifies to robust streaming tensor factorization if Ω_t includes all possible indices.

In order to enforce the low-rank property of $\tilde{\boldsymbol{\mathcal{X}}}_t$, we assume the following CP representation for $t \in [T - i + 1, T]$:

$$\tilde{\boldsymbol{\mathcal{X}}}_t = [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}; \hat{\mathbf{a}}_{t-i+1}^{(N+1)}]\!].$$

The sparsity of $\boldsymbol{\mathcal{S}}_t$ can be enforced by adding an L_1 regularizer and modifying (4) as follows:

$$\begin{aligned} \min_{\{\mathbf{A}^{(j)}\}, \boldsymbol{\mathcal{S}}_{\Omega_T}} & \sum_{t=i}^{T-1} \mu^{T-t} \| \left(\tilde{\boldsymbol{\mathcal{D}}}_t - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}; \hat{\mathbf{a}}_{t-i+1}^{(N+1)}]\!] \right)_{\Omega_t} \|_{\mathrm{F}}^2 \\ & + \| \boldsymbol{\mathcal{Y}}_{\Omega_T} - \boldsymbol{\mathcal{S}}_{\Omega_T} - \left([\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}; \hat{\mathbf{a}}_{T-i+1}^{(N+1)}]\!] \right)_{\Omega_T} \|_{\mathrm{F}}^2 \\ & + \alpha \| \boldsymbol{\mathcal{S}}_{\Omega_T} \|_1. \end{aligned} \quad (6)$$

Here $\boldsymbol{\mathcal{Y}}_{\Omega_T} = \boldsymbol{\mathcal{X}}_{T,\Omega_T}$ is the observation of current slice, $\boldsymbol{\mathcal{S}}_{\Omega_T}$ is its outliers, and $\{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\}_{t=i}^{T-1}$ are the observed past slices *with their sparse errors removed*. Based on the results of of all previous slices, $\tilde{\boldsymbol{\mathcal{D}}}_t$ is obtained as $\tilde{\boldsymbol{\mathcal{D}}}_t = \boldsymbol{\mathcal{X}}_t - \boldsymbol{\mathcal{S}}_t$.

It is challenging to determine the rank R in (6). It is also non-trivial to select a proper regularization parameter α. In order to fix these issues, we develop a Bayesian model.

B. Probabilistic Model for (5)

Likelihood: We first define a likelihood function for the data $\boldsymbol{\mathcal{Y}}_{\Omega_T}$ and $\{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\}_{t=i}^{T-1}$ based on (5) and (6). We discount the past observations outside of the time window, and use the forgetting factor $\mu < 1$ to exponentially weight the variance terms of past observations. This permits long-past observations to deviate significantly from the current CP factors with little impact on the current CP factors. Therefore, at time point t, we assume that the Gaussian noise has a 0 mean and variance $(\mu^{T-t}\tau)^{-1}$. This leads to the likelihood function in (7). In

$$p\left(\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\}\middle|\{\mathbf{A}^{(n)}\}_{n=1}^{N+1}, \boldsymbol{\mathcal{S}}_{\Omega_T}, \tau\right) = \prod_{(i_1,\ldots,i_n)\in\Omega_T} \mathcal{N}\left(\boldsymbol{\mathcal{Y}}_{i_1\ldots i_N}\middle|\left\langle \hat{\mathbf{a}}_{i_1}^{(1)}, \cdots, \hat{\mathbf{a}}_{i_N}^{(N)}, \hat{\mathbf{a}}_{T-i+1}^{(N+1)}\right\rangle + \mathcal{S}_{i_1\ldots i_N}, \tau^{-1}\right) \times \prod_{t=i}^{T-1}\prod_{(i_1,\ldots,i_n)\in\Omega_t} \mathcal{N}\left(\tilde{\boldsymbol{\mathcal{D}}}_{t,i_1\ldots i_N}\middle|\left\langle \hat{\mathbf{a}}_{i_1}^{(1)}, \cdots, \hat{\mathbf{a}}_{i_N}^{(N)}, \hat{\mathbf{a}}_{t-i+1}^{(N+1)}, \right\rangle, (\tau\mu^{T-t})^{-1}\right). \quad (7)$$

$$p\left(\Theta\middle|\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\}\right) = \frac{p\left(\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\}\middle|\ \{\mathbf{A}^{(n)}\}_{n=1}^{N+1}, \boldsymbol{\mathcal{S}}_{\Omega_T}, \tau\right)\left\{\prod_{n=1}^{(N+1)} p\big(\mathbf{A}^{(n)}\big|\boldsymbol{\lambda}\big)\right\} p(\boldsymbol{\lambda})p(\boldsymbol{\mathcal{S}}_{\Omega_T}|\boldsymbol{\gamma})p(\boldsymbol{\gamma})p(\tau)}{p(\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\})}. \quad (10)$$

this likelihood function, τ specifies the noise precision, $\hat{\mathbf{a}}_{i_n}^{(n)}$ denotes the i_n-th row of $\mathbf{A}^{(n)}$, and $\boldsymbol{\mathcal{S}}_{\Omega_T}$ only has values corresponding to observed locations. In order to infer the unknown factors, we also specify their prior distributions.

Prior Distribution of $\{\mathbf{A}^{(n)}\}$: We assume that each row of $\mathbf{A}^{(n)}$ obeys a Gaussian distribution and that different rows are independent to each other. Similar to [19], we define the prior distribution of each factor matrix as

$$p\big(\mathbf{A}^{(n)}\big|\boldsymbol{\lambda}\big) = \prod_{i_n=1}^{I_n} \mathcal{N}\big(\hat{\mathbf{a}}_{i_n}^{(n)}\big|\mathbf{0}, \boldsymbol{\Lambda}^{-1}\big),\ \forall n \in [1, N+1] \quad (8)$$

where $\boldsymbol{\Lambda} = \text{diag}(\boldsymbol{\lambda}) \in \mathbb{R}^{R\times R}$ denotes the precision matrix. All factor matrices share the same covariance matrix. The r-th column of all factor matrices share the same precision parameter λ_r, and a large λ_r will make the r-th rank-1 term more likely to have a small magnitude. Therefore, the hyper parameters $\boldsymbol{\lambda} \in \mathbb{R}^R$ can tune the rank of our CP model.

Prior Distribution of $\boldsymbol{\mathcal{S}}_{\Omega_T}$: We also place a Gaussian prior distribution over the component $\boldsymbol{\mathcal{S}}_{\Omega_T}$:

$$p(\boldsymbol{\mathcal{S}}_{\Omega_T}|\boldsymbol{\gamma}) = \prod_{(i_1,\ldots,i_N)\in\Omega_T} \mathcal{N}(\mathcal{S}_{i_1\ldots i_N}|0, \gamma_{i_1\ldots i_N}^{-1}), \quad (9)$$

where $\boldsymbol{\gamma}$ is the sparsity precision matrix. If $\gamma_{i_1\ldots i_N}$ is very large, then the associated element in $\boldsymbol{\mathcal{S}}_{\Omega_T}$ is likely to have a very small magnitude. By controlling the value of $\gamma_{i_1\ldots i_N}^{-1}$, we can control the sparsity of $\boldsymbol{\mathcal{S}}_{\Omega_T}$.

C. Prior Distribution of Hyper Parameters

We still have to specify three groups of hyper parameters: τ controlling the noise term, $\boldsymbol{\lambda}$ controlling the CP rank, and $\{\gamma_{i_1\ldots i_N}\}$ controlling the sparsity of $\boldsymbol{\mathcal{S}}_{\Omega_T}$. We treat them as random variables and assign them Gamma prior distributions:

$$\begin{aligned} &p(\tau) = \text{Ga}(\tau \mid a_0^\tau, b_0^\tau),\ \ p(\boldsymbol{\lambda}) = \prod_{r=1}^{R} \text{Ga}(\lambda_r|c_0, d_0), \\ &p(\boldsymbol{\gamma}) = \prod_{(i_1,\ldots,i_N)\in\Omega_T} \text{Ga}(\gamma_{i_1\ldots i_N}|a_0^\gamma, b_0^\gamma). \end{aligned} \quad (11)$$

The Gamma distribution provides a good model for our hyper parameters due to its non-negativity and its long tail. The mean value and variance of the above Gamma distribution are a/b and a/b^2, respectively.

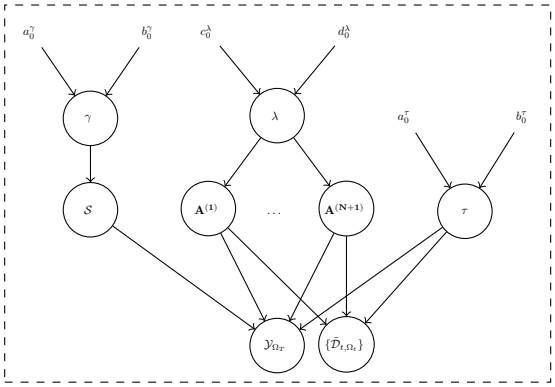

Fig. 1. The probabilistic graphical model for our Bayesian robust streaming tensor completion.

These hyper parameters control $\{\mathbf{A}^{(n)}\}$ and $\boldsymbol{\mathcal{S}}$. For instance, the noise term tends to have a very small magnitude if τ has a large mean value and a small variance; if λ_r has a large mean value, then the r-th rank-1 term in the CP factorization tends to vanish, leading to rank reduction.

D. Posterior Distribution of Model Parameters

Now we can present a graphical model describing our Bayesian formulation in Fig. 1. Our goal is to infer all hidden parameters based on partially observed data. For convenience, we denote all unknown hidden parameters in a compact form:

$$\Theta = \left\{\{\mathbf{A}^{(n)}\}_{n=1}^{N+1}, \boldsymbol{\mathcal{S}}_{\Omega_T}, \tau, \boldsymbol{\lambda}, \boldsymbol{\gamma}\right\}.$$

With the likelihood function (7), prior distribution for low-rank factors and sparse components in (8) and (9), and prior distribution of the hyper-parameters in (11), we can obtain the posterior distribution in (10) using Bayes theorem.

The main challenge is how to estimate the resulting posterior distribution (10). We address this issue in Section V.

V. Variational Bayesian Solver For Model Parameter Estimation

It is hard to obtain the exact posterior distribution (10) because the marginal density $p(\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\})$ is unknown and is expensive to compute. Therefore, we employ variational

Algorithm 1 Variational Bayesian Updating Scheme for Streaming Tensor Completion

while Not Converged **do**
  Update the variance matrices via Equations (15,17)
  Update the factor matrices by Equations (16,18, 19)
  Update the rank prior $\boldsymbol{\lambda}$ by Equation (20)
  Update the sparse term $\boldsymbol{\mathcal{S}}_{\Omega_T}$ by Equation (22)
  Update the sparsity prior $\boldsymbol{\gamma}$ by Equation (24)
  Update the precision τ by Equation (25)
end while

Bayesian inference [21] to obtain a closed-form approximation of the posterior density (10). The variational Bayesian method was previously employed for matrix completion [22] and non-streaming tensor completion [19], [20], and it is a popular inference technique in many domains. We use a similar procedure to [19], [22] to derive our iteration steps, but the details are quite different since we solve a streaming problem and we approximate an entirely different posterior distribution.

Due to the complexity of our analysis and the page space limitation, we provide only the main idea and key results of our solver. The algorithm flow is summarized in Alg. 1, and the complete derivations are available in [23].

A. Variational Bayesian

We intend to find a distribution $q(\Theta)$ that approximates the true posterior distribution $p(\Theta|\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\})$ by minimizing the following KL divergence:

$$\mathrm{KL}\big(q(\Theta)\big|\big|p(\Theta|\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\})\big) = \ln p(\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\}) - \mathcal{L}(q),$$
$$\text{where } \mathcal{L}(q) = \int q(\Theta) \ln \left(\frac{p(\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\}, \Theta)}{q(\Theta)} \right) d\Theta. \tag{12}$$

The quantity $\ln p(\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\})$ denotes model evidence and is a constant. Therefore, minimizing the KL divergence is equivalent to maximizing $\mathcal{L}(q)$. To do so we apply mean field variational approximation. That is, we assume that the posterior can be factorized as a product of the individual marginal distributions:

$$q\left(\Theta\right) = \left\{ \prod_{n=1}^{N+1} q\left(\mathbf{A}^{(n)}\right) \right\} q(\boldsymbol{\mathcal{S}}_{\Omega_T}) q(\boldsymbol{\lambda}) q(\boldsymbol{\gamma}) q(\tau). \tag{13}$$

This assumption allows us to maximize $\mathcal{L}(q)$ by applying the following update rule

$$\ln q(\Theta_i) \propto \max_{\Theta_i} \mathbb{E}_{\Theta_{j\neq i}} \ln(p(\boldsymbol{\mathcal{Y}}_{\Omega_T}, \{\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t}\}, \Theta)), \tag{14}$$

where the subscript $\Theta_{j\neq i}$ denotes the expectation with respect to all latent factors except Θ_i. In the following we will provide the closed-form expressions of these alternating updates.

B. Factor Matrix Updates

The posterior distribution of an individual factor matrix is

$$q\big(\mathbf{A}^{(n)}\big) = \prod_{i_n=1}^{I_n} \mathcal{N}\big(\hat{\mathbf{a}}_{i_n}^{(n)} \big| \bar{\mathbf{a}}_{i_n}^{(n)}, \mathbf{V}_{i_n}^{(n)}\big).$$

Therefore, we must update the posterior mean $\bar{\mathbf{a}}_{i_n}^{(n)}$ and covariance $\mathbf{V}_{i_n}^{(n)}$ for each row of $\mathbf{A}^{(n)}$.

Update non-temporal factors. All non-time factors are updated by Equations (15) and (16) for $n \in [1, \ldots, N]$:

$$\mathbf{V}_{i_n}^{(n)} = \left(\mathbb{E}_q[\tau] \sum_{t=i}^{T} \mu^{T-i} \mathbb{E}_q \left[\mathbf{A}_{i_n}^{(\backslash n)T} \mathbf{A}_{i_n}^{(\backslash n)} \right]_{\Omega_t} + \mathbb{E}_q[\boldsymbol{\Lambda}] \right)^{-1}, \tag{15}$$

$$\bar{\mathbf{a}}_{i_n}^{(n)} = \mathbb{E}_q[\tau] \mathbf{V}_{i_n}^{(n)} \Bigg(\mathbb{E}_q \left[\mathbf{A}_{i_n}^{(\backslash n)T} \right]_{\Omega_T} \mathrm{vec}\left(\boldsymbol{\mathcal{Y}}_{\Omega_T} - \mathbb{E}_q[\boldsymbol{\mathcal{S}}_{\Omega_T}] \right) + \sum_{t=i}^{T-1} \mu^{T-t} \mathbb{E}_q \left[\mathbf{A}_{i_n}^{(\backslash n)T} \right]_{\Omega_t} \mathrm{vec}\left(\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t} \right) \Bigg). \tag{16}$$

The notation $\mathbb{E}_q \left[\mathbf{A}_{i_n}^{(\backslash n)} \right]_{\Omega_t}$ represents a sampled expectation of the excluded Khatri-Rao product:

$$\mathbb{E}_q \left[\mathbf{A}_{i_n}^{(\backslash n)} \right]_{\Omega_t} = \left(\mathbb{E}_q \left[\bigodot_{j\neq n} \mathbf{A}^{(j)} \right] \right)_{\mathbb{I}_{i_n}}.$$

The matrix $\mathbf{A}_{i_n}^{(\backslash n)}$ is $\prod_{j\neq n} I_j \times R$ and the indicator function $\mathbb{I}_{i_n}$ samples the row $(i_1, \ldots, i_{n-1}, i_{n+1}, \ldots, i_{N+1})$ if the entry $(i_1, \ldots, i_{n-1}, i_n, i_{n+1}, \ldots, i_{N+1})$ is in Ω_t and sets the row to zero if not. The expression $\mathbb{E}_q[\cdot]$ denotes the posterior expectation with respect to all variables involved.

Update temporal factors. The temporal factor $\mathbf{A}^{N+1}$ requires a different update because the factors corresponding to different time slices do not interact with each other. For all time factors the variance is updated according to

$$\mathbf{V}_{t-i+1}^{(N+1)} = \left(\mathbb{E}_q[\tau] \mu^{T-t} \mathbb{E}_q \left[\mathbf{A}_{t-i+1}^{(\backslash(N+1))T} \mathbf{A}_{t-i+1}^{(\backslash(N+1))} \right]_{\Omega_t} + \mathbb{E}_q[\boldsymbol{\Lambda}] \right)^{-1}. \tag{17}$$

The rows of the time factor matrix are updated differently depending on their corresponding time indices. Since we assume that past observations have had their sparse errors removed, the time factors of all past slices (so $t < T$) can be updated by

$$\bar{\mathbf{a}}_{t-i+1}^{(N+1)} = \mathbb{E}_q[\tau] \mathbf{V}_{t-i+1}^{(N+1)} \left(\mu^{T-t} \mathbb{E}_q \left[\mathbf{A}_{t-i+1}^{(\backslash N+1)T} \right]_{\Omega_t} \mathrm{vec}\left(\tilde{\boldsymbol{\mathcal{D}}}_{t,\Omega_t} \right) \right). \tag{18}$$

The factors corresponding to time slice T depend on the sparse errors in the current step. The update is therefore given by

$$\bar{\mathbf{a}}_{T-i+1}^{(N+1)} = \mathbb{E}_q[\tau] \mathbf{V}_{T-i+1}^{(N+1)} \left(\mathbb{E}_q \left[\mathbf{A}_{T-i+1}^{(\backslash N+1)T} \right]_{\Omega_T} \mathrm{vec}\left(\boldsymbol{\mathcal{Y}}_{\Omega_T} - \mathbb{E}_q\left[\boldsymbol{\mathcal{S}}_{\Omega_T} \right] \right) \right). \tag{19}$$

C. Posterior Distribution of Hyperparameters $\boldsymbol{\lambda}$

The posteriors of the parameters λ_r are independent Gamma distributions. Therefore the joint distribution takes the form

$$q(\boldsymbol{\lambda}) = \prod_{r=1}^{R} \mathrm{Ga}(\lambda_r | c_M^r, d_M^r)$$

where c_M^r, d_M^r denote the posterior parameters learned from the previous M iterations. The updates to $\boldsymbol{\lambda}$ are given below.

$$c_M^r = c_0 + 1 + \frac{1}{2}\sum_{n=1}^{N} I_n, \quad d_M^r = d_0 + \frac{1}{2}\sum_{n=1}^{N+1} \mathbb{E}_q\left[\mathbf{a}_r^{(n)T}\mathbf{a}_r^{(n)}\right] \tag{20}$$

The expectation of each rank-sparsity parameter can then be computed as

$$\mathbb{E}_q[\boldsymbol{\Lambda}] = \text{diag}([c_M^1/d_M^1, \ldots, c_M^R/d_M^R]).$$

D. Posterior Distribution of Sparse tensor $\boldsymbol{\mathcal{S}}$

The posterior approximation of $\boldsymbol{\mathcal{S}}_{\Omega_T}$ is given by

$$q(\boldsymbol{\mathcal{S}}_{\Omega_T}) = \prod_{(i_1,\ldots,i_N)\in\Omega_T} \mathcal{N}\left(\mathcal{S}_{i_1\ldots i_N} \middle| \bar{\mathcal{S}}_{i_1\ldots i_N}, \sigma^2_{i_1\ldots i_N}\right), \tag{21}$$

where the posterior parameters can be updated by

$$\begin{aligned}\bar{\mathcal{S}}_{i_1\ldots i_N} =& \sigma^2_{i_1\ldots i_N}\mathbb{E}_q[\tau]\Big(\mathcal{Y}_{i_1\ldots i_N} - \\ & \mathbb{E}_q\left[\left\langle \hat{\mathbf{a}}_{i_1}^{(1)}, \ldots, \hat{\mathbf{a}}_{i_N}^{(N)}; \hat{\mathbf{a}}_{T-i+1}^{(N+1)}\right\rangle\right]\Big) \\ \sigma^2_{i_1\ldots i_N} =& (\mathbb{E}_q[\gamma_{i_1\ldots i_N}] + \mathbb{E}_q[\tau])^{-1}.\end{aligned} \tag{22}$$

E. Posterior Distribution of Hyperparameters $\boldsymbol{\gamma}$

The posterior of $\boldsymbol{\gamma}$ is also factorized into entry-wise independent distributions

$$q(\boldsymbol{\gamma}) = \prod_{(i_1,\ldots,i_N)\in\Omega_T} \text{Ga}(\gamma_{i_1\ldots i_N} | a_M^{\gamma_{i_1\ldots i_N}}, b_M^{\gamma_{i_1\ldots i_N}}), \tag{23}$$

whose posterior parameters can be updated by

$$a_M^{\gamma_{i_1\ldots i_N}} = a_0^\gamma + \frac{1}{2},\ b_M^{\gamma_{i_1\ldots i_N}} = b_0^\gamma + \frac{1}{2}(\bar{\mathcal{S}}^2_{i_1\ldots i_N} + \sigma^2_{i_1\ldots i_N}). \tag{24}$$

F. Posterior Distribution of Parameter τ

The posterior PDF of the noise precision is again a Gamma distribution. The posterior parameters can be updated by

$$\begin{aligned}a_M^\tau =& \frac{1}{2}|\Omega_T| + a_0^\tau, \\ b_M^\tau =& \frac{1}{2}\mathbb{E}_q\left[\| \left(\boldsymbol{\mathcal{Y}} - \boldsymbol{\mathcal{S}} - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}; \hat{\mathbf{a}}_{T-i+1}^{(N+1)}]\!]\right)_{\Omega_T} \|_{\text{F}}^2\right] \\ & + b_0^\tau.\end{aligned} \tag{25}$$

VI. Numerical Results

Our algorithm is implemented in Matlab and is compared with several existing streaming tensor factorization and completion methods. These include Online-CP [15], Online-SGD [16] and OLSTEC [17]. Both Online-CP and OLSTEC solve essentially the same optimization problem, but Online-CP does not support incomplete tensors. Therefore, our algorithm is only compared with OLSTEC and Online-SGD for the completion task. Our Matlab codes to reproduce all figures and results can be downloaded from https://github.com/colehawkins. We provide more extensive numerical results, including results on surveillance video and automatic rank determination, in [23].

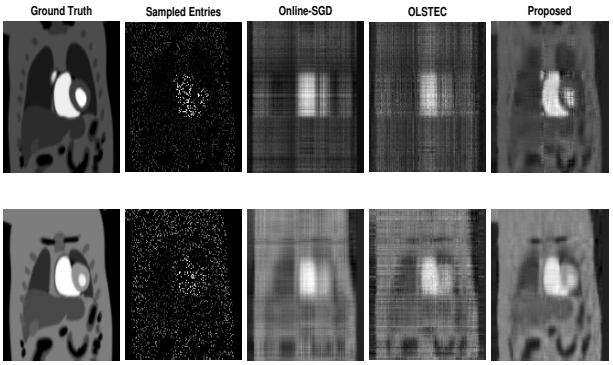

Fig. 2. MRI reconstruction via streaming tensor completion.

A. Dynamic Cardiac MRI

We consider a dynamic cardiac MRI dataset from [24] and obtained via https://statweb.stanford.edu/~candes/SURE/data.html. Each slice of this streaming tensor dataset is a 128×128 matrix. In clinical applications, it is highly desirable to reduce the number of MRI scans. Therefore, we are interested in using streaming tensor completion to reconstruct the whole sequence of medical images based on a few sampled entries.

In all methods we set the maximum rank to 15. For our algorithm we set the forgetting factor to $\mu = 0.98$ and the the sliding window size to 20. In OLSTEC we set the forgetting factor to the suggested default of 0.7 and the sliding window size to 20. The available implementation of Online-SGD does not admit a sliding window, but instead computes with the full (non-streamed) tensor. While this may limit its ability to work with large streamed data in practice, we include it in comparison for completeness. With 15% random samples, the reconstruction results are shown in Fig. 2. The ability of our model to capture both small-magnitude measurement noise and sparse large-magnitude deviations renders it more effective than OLSTEC and Online-SGD for this dynamic MRI reconstruction task.

B. Network Traffic

Our next example is the Abilene network traffic dataset [25]. This dataset consists of aggregate Internet traffic between 11 nodes, measured at five-minute intervals. On this dataset we test our algorithm for both reconstruction and completion. The goal is to identify abnormally evolving network traffic patterns between nodes. If one captures the underlying low-rank structure, one can identify anomalies for further inspection. Anomalies can range from malicious distributed denial of service (DDoS) attacks to non-threatening network traffic spikes related to online entertainment releases. In order to classify abnormal behavior one must first fit the existing data. We evaluate the accuracy of the models under comparison by calculating the relative prediction error at each time slice:

$$\|\boldsymbol{\mathcal{X}}_t - [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}, \hat{\mathbf{a}}_t^{(N+1)}]\!] - \boldsymbol{\mathcal{S}}_t\|_{\text{F}} / \|\boldsymbol{\mathcal{X}}_t\|_{\text{F}}.$$

Fig. 3 compares different methods on the full dataset with a

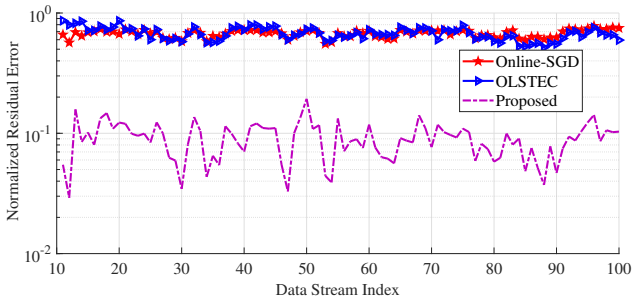

Fig. 3. Factorization error of network traffic from complete samples.

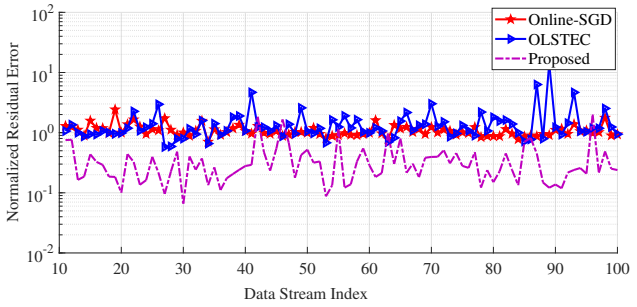

Fig. 4. Reconstruction error of network traffic with 50% of data missing.

"burn-in" time of 10 frames, after which the error patterns are stable. Our algorithm significantly outperforms OLSTEC and Online-SGD in factoring the whole data set. Then we remove 50% of the entries from the the Abilene tensor and attempt to reconstruct the whole network traffic. Our results are shown in Fig. 4. Again we use a "burn-in" time of 10 frames.

VII. Conclusion

We have proposed a Bayesian formulation to the problem of robust streaming tensor completion and factorization. The main advantages of our algorithm are **automatic rank determination** and **robustness** to outliers. We have demonstrated the benefits of robustness on MRI and network flow data. Due to the automatic rank determination and the robustness to outliers, our algorithm has achieved higher accuracy than existing approaches on all tested streaming tensor examples.

VIII. Acknowledgements

We thank the anonymous referees for their helpful comments. A special thanks to Chunfeng Cui for many suggestions to improve this manuscript. This work was partially supported by NSF CCF Award No. 1817037.

References

[1] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering," in *Proc. ACM Conf. Recommender systems*, 2010, pp. 79–86.

[2] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 208–220, 2013.

[3] Z. Zhang, T.-W. Weng, and L. Daniel, "Big-data tensor recovery for high-dimensional uncertainty quantification of process variations," *IEEE Trans. Components, Packaging and Manufacturing Technology*, vol. 7, no. 5, pp. 687–697, 2017.

[4] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[5] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an N-way generalization of eckart-young decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[6] R. A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis," 1970.

[7] J. Zhou, A. Bhattacharya, A. H. Herring, and D. B. Dunson, "Bayesian factorizations of big sparse tensors," *Journal of the American Statistical Association*, vol. 110, no. 512, pp. 1562–1576, 2015.

[8] P. Jain and S. Oh, "Provable tensor factorization with missing data," in *Advances in Neural Information Processing Systems*, 2014, pp. 1431–1439.

[9] D. Kressner, M. Steinlechner, and B. Vandereycken, "Low-rank tensor completion by Riemannian optimization," *BIT Numerical Mathematics*, vol. 54, no. 2, pp. 447–468, 2014.

[10] H. Fanaee-T and J. Gama, "Tensor-based anomaly detection: An interdisciplinary survey," *Knowledge-Based Systems*, vol. 98, pp. 130–147, 2016.

[11] B. W. Bader, M. W. Berry, and M. Browne, "Discussion tracking in enron email using parafac," in *Survey of Text Mining II*. Springer, 2008, pp. 147–163.

[12] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering," in *Proc. ACM Conf. Recommender systems*, 2010, pp. 79–86.

[13] J. Sun, D. Tao, and C. Faloutsos, "Beyond streams and graphs: dynamic tensor analysis," in *Proc. ACM SIGKDD Int. Conf. Knowledge discovery and data mining*, 2006, pp. 374–383.

[14] S. Smith, K. Huang, N. D. Sidiropoulos, and G. Karypis, "Streaming tensor factorization for infinite data sources," in *Proc. SIAM Int. Confe. Data Mining*, 2018, pp. 81–89.

[15] S. Zhou, N. X. Vinh, J. Bailey, Y. Jia, and I. Davidson, "Accelerating online CP decompositions for higher order tensors," in *Proc. ACM SIGKDD Intl. Conf. Knowledge Discovery and Data Mining*, 2016, pp. 1375–1384.

[16] M. Mardani, G. Mateos, and G. B. Giannakis, "Subspace learning and imputation for streaming big data matrices and tensors," *IEEE Transactions on Signal Processing*, vol. 63, no. 10, pp. 2663–2677, 2015.

[17] H. Kasai, "Online low-rank tensor subspace tracking from incomplete data by CP decomposition using recursive least squares," in *Int. Conf. Acoustics, Speech and Signal Processing*, 2016, pp. 2519–2523.

[18] J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma, "Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization," in *Advances in neural information processing systems*, 2009, pp. 2080–2088.

[19] Q. Zhao, L. Zhang, and A. Cichocki, "Bayesian CP factorization of incomplete tensors with automatic rank determination," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1751–1763, 2015.

[20] Q. Zhao, G. Zhou, L. Zhang, A. Cichocki, and S.-I. Amari, "Bayesian robust tensor factorization for incomplete multiway data," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 4, pp. 736–748, 2016.

[21] J. Winn and C. M. Bishop, "Variational message passing," *Journal of Machine Learning Research*, vol. 6, no. Apr, pp. 661–694, 2005.

[22] S. D. Babacan, M. Luessi, R. Molina, and A. K. Katsaggelos, "Sparse bayesian methods for low-rank matrix estimation," *IEEE Transactions on Signal Processing*, vol. 60, no. 8, pp. 3964–3977, 2012.

[23] C. Hawkins and Z. Zhang, "Robust factorization and completion of streaming tensor data via variational bayesian inference," *arXiv preprint arXiv:1809.01265*, 2018.

[24] B. Sharif and Y. Bresler, "Physiologically improved NCAT phantom (PINCAT) enables in-silico study of the effects of beat-to-beat variability on cardiac MR," in *Proc. ISMRM, Berlin*, vol. 3418, 2007.

[25] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," in *ACM SIGMETRICS Performance evaluation review*, vol. 32, no. 1, 2004, pp. 61–72.