Graph-Assisted Tensor Disaggregation

Mariana M Garcez Duarte
UC Riverside
Riverside, USA
mmach027@ucr.edu

Evangelos E. Papalexakis
UC Riverside
Riverside, USA
epapalex@cs.ucr.edu

Jia Chen
UC Riverside
Riverside, USA
jiac@ucr.edu

ABSTRACT

Consider a multi-aspect tensor dataset which is only observed in multiple complementary aggregated versions, each one at a lower resolution than the highest available one. Recent work [2] has demonstrated that given two such tensors, which have been aggregated in lower resolutions in complementary dimensions, we can pose and solve the disaggregation as an instance of a coupled tensor decomposition. In this work, we are exploring the scenario in which, in addition to the two complementary aggregated views, we also have access to a graph where nodes correspond to samples of the tensor mode that has not been aggregated. Given this graph, we propose a graph-assisted tensor disaggregation method. In our experimental evaluation, we demonstrate that our proposed method performs on par with the state of the art when the rank of the underlying coupled tensor decomposition is low, and significantly outperforms the state of the art in cases where the rank increases, producing more robust and higher-quality disaggregation.

CCS CONCEPTS

• **Computing methodologies** → **Regularization**; **Factorization methods**;

KEYWORDS

Tensor disaggregation, graph regularization, tensor decomposition, data preprocessing

ACM Reference Format:
Mariana M Garcez Duarte, Evangelos E. Papalexakis, and Jia Chen. 2022. Graph-Assisted Tensor Disaggregation. In *Proceedings of 17th International Workshop on Mining and Learning with Graphs (MLG '22).* ACM, New York, NY, USA, 5 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Consider as our running example a multi-aspect dataset that records the number of scientific paper publications per publisher (e.g., IEEE, ACM, SIAM, and Springer) over time and by institution. This can be represented as a (publisher, time, university) tensor, and analyzing such a tensor may be able to offer valuable insights to publication trends over time and space, which can be beneficial to universities and publishers alike.

Consider, further, the scenario in which different stakeholders may observe varying pieces of the data in terms of resolution. For instance, stakeholder #1 may be able to observe publication counts from different universities on a yearly basis as opposed to a more fine-grained monthly basis, resulting in a (publisher, year, university) tensor. At the same time, stakeholder #2 may be able to observe monthly publication counts per publisher per country, but lack the finer university-level granularity, resulting in a (publisher, month, country) tensor.

Those two stakeholders essentially have access to two complementary aggregated views of the ideal high resolution tensor. How can we effectively combine those two views in order to recover the dataset in the highest available resolution in all its modes? Recently, a novel tensor method, namely PREMA, was proposed to to solve such a tensor disaggregation problem [1; 2], by posing the problem as a variant of coupled tensor decomposition.

In this paper, following the example above, we focus on the following question: "*If we also have access to a (publisher, publisher) graph, can we improve disaggregation performance?*". More specifically, given such a graph, we extend PREMA [1; 2] to account for graph Laplacian smoothness in the factor matrix corresponding to the mode for which we have a graph available. Essentially, we are ensuring that the rows of that matrix, which, in our running example are publisher "embeddings", are encouraged to obey publisher similarities defined by our graph. This graph can be either directly computed from the aggregated data or provided by a domain expert. In this preliminary work, we directly compute graphs from the aggregated data.

We conduct experiments on two real-world datasets and we measure the behavior of our proposed method. We center our analysis on an important parameter of the problem which is the rank of the decomposition, which dictates the fidelity of the reconstruction, since a higher-rank decomposition represents the data more accurately than a lower-rank one. We observe that our proposed graph assisted tensor disaggregation method performs on par with state of the art for relatively low ranks. However, as the rank increases, our proposed method significantly outperforms state of the art, providing higher fidelity disaggregation while also increasing the robustness of the disaggregation process to potential overestimations of the "optimal" rank for a given dataset.

2 BACKGROUND

First, we will review the basis of tensor algebra. Tensors are arrays having three or more than three dimensions with indices $(i, j, k, \ldots)$. For the sake of simplicity, we focus on the three-dimensional (a.k.a., third-order or three-mode) tensors. Let's denote a general tensor $\mathcal{Z} \in \mathbb{R}^{I \times J \times K}$ consisting of three modes: rows $\mathcal{Z}(:, j, k)$, columns $\mathcal{Z}(i, :, k)$, and fibers $\mathcal{Z}(i, j, :)$. The horizontal, lateral, and frontal slabs of $\mathcal{Z}$ are denoted by $\mathcal{Z}(i, :, :)$, $\mathcal{Z}(:, j, :)$, and

MLG '22, August 15, 2022, Washington, DC

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

$\mathcal{Z}(:, :, k)$, respectively, for $i = 1, ..., I$, $j = 1, ..., J$, and $k = 1, ..., K$. Tensor decomposition is the core technique for many tensor-based methods and the commonly used models are Canonical Polyadic Decomposition (CPD) (a.k.a., PARAFAC) and Tucker decomposition [5]. Next, we briefly introduce the CPD model which factors a tensor into the sum of three-way outer products, i.e.,

$$\mathcal{Z} \approx \sum_{r=1}^{R} \mathsf{a}_r \circ \mathsf{b}_r \circ \mathsf{c}_r \tag{1}$$

where R is the rank of the tensor, $\mathsf{a}_r \in \mathbb{R}^I$, $\mathsf{b}_r \in \mathbb{R}^J$, $\mathsf{c}_r \in \mathbb{R}^K$, and the three-way outer product is defined as $(\mathsf{a}_r \circ \mathsf{b}_r \circ \mathsf{c}_r)(i, j, k) := \mathsf{a}_r(i)\mathsf{b}_r(j)\mathsf{c}_r(k)$. The factor matrices denoted by $\mathsf{A} \in \mathbb{R}^{I \times R}$, $\mathsf{B} \in \mathbb{R}^{J \times R}$, and $\mathsf{C} \in \mathbb{R}^{K \times R}$ collect a_r, b_r, and c_r, in their columns, i.e.,

$$\mathsf{A} := [\mathsf{a}_1, ..., \mathsf{a}_R], \quad \mathsf{B} := [\mathsf{b}_1, ..., \mathsf{b}_R], \quad \mathsf{C} := [\mathsf{c}_1, ..., \mathsf{c}_R]. \tag{2}$$

We can compactly represent the CPD of a tensor into factors A, B and C as $[[\mathsf{A}, \mathsf{B}, \mathsf{C}]]$.

3 PROPOSED METHOD

Consider two aggregated tensors, namely $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$, which are seen as the two views of the original tensor $\mathcal{Z}$, obtained from the mode product between $\mathcal{Z}$ and the aggregation matrices. Specifically, $\boldsymbol{\mathcal{X}} := \mathcal{Z} \times_3 \mathsf{W} \in \mathbb{R}^{I \times J \times D}$ where $\mathsf{W} \in \mathbb{R}^{D \times K}$ represents an aggregation matrix with D_K Ÿ K and $\times_3$ denotes the mode-3 product operator. Without loss of generality, another view is aggregated from both the first and second dimensions of $\boldsymbol{\mathcal{X}}$, that is $\boldsymbol{\mathcal{Y}} := \mathcal{Z} \times_1 \mathsf{U} \times_2 \mathsf{V} \in \mathbb{R}^{D \times D \times K}$ with D_I Ÿ I, D_J Ÿ J, $\mathsf{U} \in \mathbb{R}^{D \times I}$, and $\mathsf{V} \in \mathbb{R}^{D \times J}$. When the aggregation doesn't occur in either the first or second dimension of $\boldsymbol{\mathcal{Y}}$, one can simply set V or U to be an identity matrix. Recently, a novel algorithm, namely PREMA, was proposed to reconstruct the original tensor $\mathcal{Z}$ given the aggregated tensors $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$, and the aggregation matrices W, U, and V [2].

Motivated by PREMA and going beyond it, we introduce a new graph-assisted tensor disaggregation framework by leveraging the the advances of graph auxiliary knowledge in preserving the data structure. In many real applications, besides feature data, one is often able to have access to its sample-to-sample interaction graph. Following the running example of the introduction, we may be able to obtain publisher-publisher relations which can be measured in a variety of ways, including number of citations between the two publishers, number of common authors between two publishers and so on.

Such graph data can be provided by some domain experts or calculated from feature data. Exploiting such graph information in many machine learning models such as canonical correlation analysis and tensor decomposition has shown improvement in their downstream tasks [4; 6–10].

As a proof of concept, in this work we generate the graph from the aggregated data directly, and we reserve the investigation of other types of graphs for future work. Given two views of aggregated tensors $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$, we first generate a graph similarity matrix, namely $\mathsf{S} \in \mathbb{R}^{I \times I}$, from $\boldsymbol{\mathcal{X}}$ indicating the interactions between pairs of horizontal slabs. Specifically, we treat each of the slab as one node of a graph and vectorize the corresponding matrix and then calculate the linear or nonlinear similarity between any two node vectors using the kernel methods, k-nearest neighbors, and etc.

Algorithm 1: Graph-assisted tensor disaggregation

1: **Input:** aggregated tensors $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$; aggregation matrices W, U, and V; learning rates α, β, and γ; graph regularization coefficient μ; rank R; graph similarity matrix S.
2: **Calculate** graph Laplacian L.
3: **Repeat**
   Update A via Eq. 5
   Update B via Eq. 6
   Update C via Eq. 7
4: **Until** the objective is below a threshold or the number of iterations is beyond another threshold.
5: **Output:** factor matrices $\mathsf{A}, \mathsf{B}, \mathsf{C}$.

Next, we develop our new model by introducing the coupled tensor decomposition while incorporating the graph regularizer in the latent component matrix A, that is

$$\min_{\mathsf{A},\mathsf{B},\mathsf{C}} \|\boldsymbol{\mathcal{X}} - [[\mathsf{A}, \mathsf{B}, \mathsf{WC}]]\|_F^2 + \|\boldsymbol{\mathcal{Y}} - [[\mathsf{UA}, \mathsf{VB}, \mathsf{C}]]\|_F^2 + \mu \mathrm{Tr}(\mathsf{A}^\top \mathsf{LA}) \tag{3}$$

where $\mathsf{L} := \mathsf{D} - \mathsf{S} \in \mathbb{R}^{I \times I}$ denotes the graph Laplacian with D being the degree matrix, the first two terms are the tensor decomposition errors from the two views and the last term promotes graph smoothness, i.e., if two nodes, say m and n, in the graph captured by the similarity matrix S are close (i.e., the (m, n)th entry of S is high) then their corresponding latent component vectors (the m-th and n-th rows of A) are close in the Euclidean space.

The optimization problem in Eq. (3) is non-convex and NP-hard in general. To solve it, similar to PREMA, we use a Block Coordinate Descent (BCD) algorithm which alternatively updates one variable while fixing the others. After denoting the objective in Eq. (3) as f and deriving the partial derivatives of f w.r.t. A, B, and C in Eq. (4) where $\odot$ is Khatri-Rao product (a.k.a., column-wise Kronecker), $\{\boldsymbol{\mathcal{X}}_t\}$ and $\{\boldsymbol{\mathcal{Y}}_t\}$ are mode-t unfolding of the corresponding tensors, and $\top$ is matrix transpose, gradient descent technique is adopted to update the variables in each iteration; see details in Eq.s (5), (6) and (7) where α ¡ 0, β ¡ 0, and γ ¡ 0 represent the learning rates. The proposed framework is summarized in Alg. 1.

$$\begin{aligned}
\frac{\partial f}{\partial \mathsf{A}} &= 2[((\mathsf{WC}) \odot \mathsf{B})\mathsf{A}^\top - \boldsymbol{\mathcal{X}}_1]^\top ((\mathsf{WC}) \odot \mathsf{B}) \\
&\quad + 2\mathsf{U}^\top [(\mathsf{C} \odot (\mathsf{VB}))(\mathsf{UA})^\top - \boldsymbol{\mathcal{Y}}_1]^\top (\mathsf{C} \odot (\mathsf{VB})) + 2\mu \mathsf{LA} \\
\frac{\partial f}{\partial \mathsf{B}} &= 2[((\mathsf{WC}) \odot \mathsf{A})\mathsf{B}^\top - \boldsymbol{\mathcal{X}}_2]^\top ((\mathsf{WC}) \odot \mathsf{A}) \\
&\quad + 2\mathsf{V}^\top [(\mathsf{C} \odot (\mathsf{UA}))(\mathsf{VB})^\top - \boldsymbol{\mathcal{Y}}_2]^\top (\mathsf{C} \odot (\mathsf{UA})) \\
\frac{\partial f}{\partial \mathsf{C}} &= 2\mathsf{W}^\top [(\mathsf{B} \odot \mathsf{A})(\mathsf{WC})^\top - \boldsymbol{\mathcal{X}}_3](\mathsf{B} \odot \mathsf{A}) \\
&\quad + 2[((\mathsf{VB}) \odot (\mathsf{UA}))\mathsf{C}^\top - \boldsymbol{\mathcal{Y}}_3]^\top (\mathsf{C} \odot (\mathsf{UA}))
\end{aligned} \tag{4}$$

$$\mathsf{A} = \mathsf{A} - \alpha \frac{\partial f}{\partial \mathsf{A}} \tag{5}$$

$$\mathsf{B} = \mathsf{B} - \beta \frac{\partial f}{\partial \mathsf{B}} \tag{6}$$

$$\mathsf{C} = \mathsf{C} - \gamma \frac{\partial f}{\partial \mathsf{C}} \tag{7}$$

4 EXPERIMENTAL EVALUATION

4.1 Datasets

We evaluate the proposed method using two publicly available datasets:

- Walmart dataset [1] which represents the weekly sales from 45 Walmart stores and 99 departments across 143 weeks forming the tensor (stores, departments, weeks) of size $45 \times 99 \times 143$
- Indian Pines dataset [2] consisting of 145×145 pixels and 220 spectral bands which was collected from AVIRIS sensor [3].

4.2 Results

In this subsection we access the performance of our proposed method in terms of the data disaggregation improvement compared to the PREMA, that is

$$\text{Improvement percentage} = \frac{\text{PREMA NDE - Our NDE}}{\text{PREMA NDE}} \tag{8}$$

where NDE abbreviates the normalized disaggregation error, i.e.,

$$\text{NDE} = \frac{\|\mathcal{Z} - \hat{\mathcal{Z}}\|_F^2}{\|\mathcal{Z}\|_F^2} \tag{9}$$

in which $\mathcal{Z}$ and $\hat{\mathcal{Z}}$ are the real and estimated tensor data and $\hat{\mathcal{Z}} = [[\mathsf{A}, \mathsf{B}, \mathsf{C}]]$.

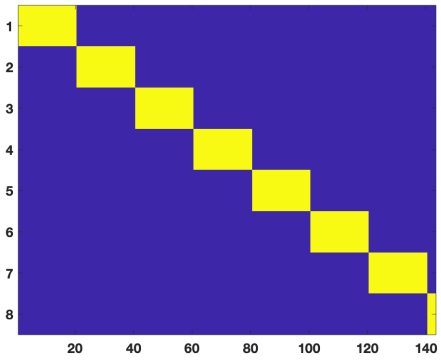

Figure 1: Aggregation matrix $\mathsf{W} \in \mathbb{R}^{8\times143}$; blue and yellow pixels are 0s and 1s; there is only one "1" in each column.

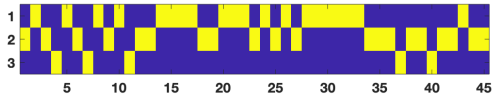

Figure 2: Aggregation matrix $\mathsf{U} \in \mathbb{R}^{3\times45}$; blue and yellow pixels are 0s and 1s; there is only one "1" in each column.

When applying the proposed method on the Walmart dataset, the aggregation matrix W is depicted in Figures 1 where almost every 7 frontal slabs in $\mathcal{Z}$ are summed to generate a new frontal slab in $\mathcal{X}$ via $\mathcal{X} = \mathcal{Z}_{\times3}\mathsf{W} \in \mathbb{R}^{I\times J\times D}$ where $I = 45$, $J = 99$, $K = 143$, $D_K = 8$, showing that the aggregated tensor $\mathcal{X}$ is only of 5.59% of the original tensor's size. Another disaggregation matrix U shown in Figure 2 allows us to randomly divide the 45 horizontal slabs of $\mathcal{Z}$ into 3 groups and sum each group to form a new horizontal slab for another disaggregated tensor $\mathcal{Y}$, i.e., $\mathcal{Y} = \mathcal{Z}_{\times1}\mathsf{U}_{\times2}\mathsf{V} \in \mathbb{R}^{D\times J\times K}$ where V is an identity matrix and $D_I = 3$. Clearly, the aggregated tensor $\mathcal{Y}$ is only of 5% of the original tensor's size. For the graph similarity matrix S, we use $\mathcal{X}$ combined with Gaussian kernel and $k-$nearest neighbors. This is realized by following four steps: 1) extract the first 20 lateral slabs of $\mathcal{X}$ forming a sub-tensor $\mathcal{X}_s \in \mathbb{R}^{45\times20\times8}$; 2) take the mode-1 matricization of the subset of $\mathcal{X}_s$ by flattening the tensor along its first mode and obtain a matrix, namely $\mathsf{X}_{s,(1)} \in \mathbb{R}^{45\times160}$; 3) apply the Gaussian kernel with bandwidth parameter 0.1 to calculate the similarity between each pair of rows in $\mathsf{X}_{s,(1)}$ and collect all the similarities to the matrix $\tilde{\mathsf{S}} \in \mathbb{R}^{45\times45}$; and 4) keep the top $k = 6$ highest values of each row in $\tilde{\mathsf{S}}$ and zero out the remaining entries forming the graph similarity matrix $\mathsf{S} \in \mathbb{R}^{45\times45}$. The learning rates α, β, and γ of our algorithm are set the in the same way as them in PREMA; see the details in Chapter 3.3 of [2]. Both our algorithm and PREMA initialize their factor matrices by performing CPD on $\mathcal{X}$ and $\mathcal{Y}$; the initial A and B come from $\mathcal{X}$ and the initial C is based on $\mathcal{Y}$.

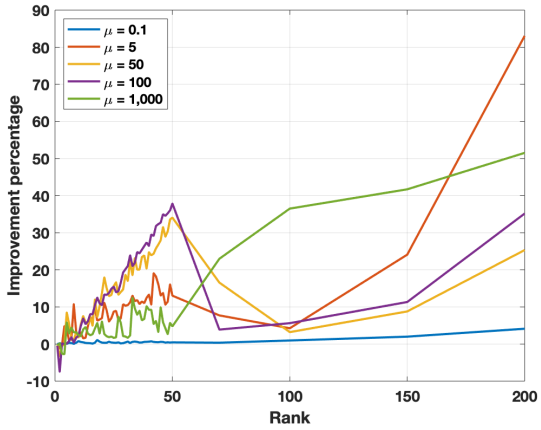

Figure 3: NDE improvement percentage of our algorithm compared to PREMA for different μs and ranks using Walmart dataset; the results show a significant data disaggregation improvement of our algorithm especially with higher ranks.

We report the average NDE improvement percentage (defined in Eq. 8) among 20 Monte Carlo runs of our algorithm w.r.t different ranks and graph regularization coefficients μs in Figure 3. In Figure 4, we plot the used μ which are chosen among the candidates 0.0001, 0.001, 0.1, 5, 15, 20, 25, 30, 50, 100, 1000 for each rank when the NDE improvement percentage reaches the highest as well as their corresponding NDE improvement percentage. From the results, we can tell that introducing graph regularizer to the PREMA increases its data disaggregation performance significantly

[1] https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting/data
[2] https://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes#Indian_Pines

Metric	Case \ R	50	100	500	1,000	1,500	2,000	3,000	4,000
Improv. Percentage	Case #1	−.1 ± .2	.5 ± 1.1	2.4 ± 1.1	1.2 ± .4	.8 ± .2	.3 ± .11	33.6 ± 8.6	**94.9 ± 6.1**
	Case #2	.2 ± .6	.5 ± .2	3.5 ± 1.1	.7 ± .1	33.7 ± 9.6	**96.5 ± 6**	**99.7 ± .2**	**98.8 ± 1.2**
	Case #3	.9 ± .8	.7 ± .3	3.8 ± .4	26.9 ± 9.8	**91.1 ± 11.3**	**95.1 ± 6.9**	**94.2 ± 7.1**	**98.8 ± 1.3**
Ours (NDE×100)	Case #1	.3 ± .01	.3 ± .05	.5 ± .05	.71 ± .04	1 ± .06	.96 ± .1	1.1 ± .2	3 ± .7
	Case #2	.4 ± .03	1.05 ± .6	1.8 ± .2	2.6 ± .2	2.4 ± .6	5.3 ± 2.3	6.4 ± 5.8	3.8 ± 1
	Case #3	.9 ± .4	2.44 ± .9	5 ± .6	2.8 ± .9	11.1 ± 7.8	8.1 ± 3.7	7 ± 2.7	6.6 ± 4.6
PREMA (NDE×100)	Case #1	.3 ± .01	.3 ± .05	.51 ± .06	.72 ± .04	1.07 ± .07	.97 ± .1	1.7 ± .4	222 ± 303
	Case #2	.4 ± .03	1.06 ± .6	1.9 ± .2	2.62 ± .2	3.8 ± 1.8	946 ± 1376	3182 ± 3001	565 ± 401
	Case #3	.9 ± .4	2.46 ± .9	5.2 ± .6	4 ± 1.9	1490 ± 5130	779 ± 1301	962 ± 1662	1398 ± 1517

Table 1: Normalized disaggregation error (NDE) comparison between our algorithm and PREMA for different ranks (R) and compression rates (a.k.a., Cases): the 2nd panel describes the NDE improvement percentage of ours compared with PREMA, and the 3rd and 4th panels report the NDE (multiplied by 100) of our algorithm and PREMA, respectively; the situations where our algorithm's improvement percentage is significant, our NDE is low, and PREMA's NDE is very high are in bold; the results show that our algorithm's performance is stable w.r.t. the rank and data compression rate.

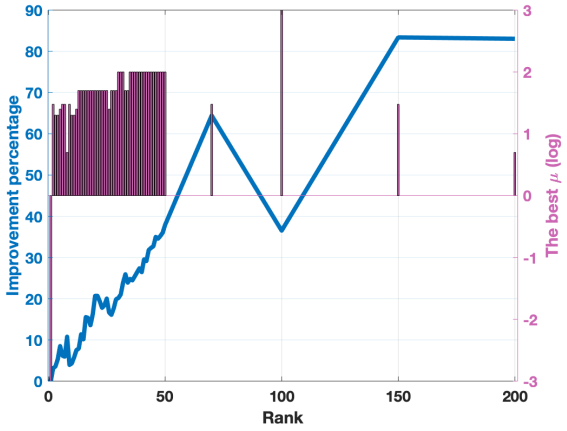

Figure 4: The highest NDE improvement percentage (the blue line in a linear scale) and the chosen μ (the purple bars in a logarithmic scale) of our algorithm v.s. rank using the Walmart dataset; the candidate μs are between 10^{-3} and 10^{3}; the results show that when μ is properly chosen our algorithm can achieve remarkably data disaggregation performance.

when μ is chosen properly, and this advantage is more apparent when the rank goes up.

Next, we investigate the effectiveness of our algorithm using the Indian Pines dataset in three cases:

Case #1: two aggregated tensors are obtained by following the similar procedure to the Walmart data aggregation but setting $D_k = 22$ and $D_I = 15$ which makes $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{145\times145\times22}$ and $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{15\times145\times220}$ of 10% and 10.34% of the original tensor size, respectively.

Case #2: similar to the Case #1 except setting $D_k = 11$ and $D_I = 8$ which makes $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{145\times145\times11}$ and $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{8\times145\times220}$ of 5% and 5.52% of the original tensor size.

Case #3: similar to the Case #1 except setting $D_k = 8$ and $D_I = 5$ which makes $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{145\times145\times8}$ and $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{5\times145\times220}$ of 3.64% and 3.45% of the original tensor size.

In Table 1, we are showing the NDE Improvement percentage of our method, the NDE of ours and PREMA for various ranks ranging from 50 to 4,000 in the above three different cases where each result is in the format of mean±standard deviation after 20 Monte Carlo experiments. Furthermore, we plot four randomly chosen samples of a single horizontal slab recovery in Figure 5 to visualize the comparison among the true, ours, and PREMA's reconstructed data. We use $\mu = 100$ for the ranks $R = 50, 100, 500$, $\mu = 1,000$ for $R = 1,000, 1,500$ and $\mu = 100,000$ for $R \geq 1,500$. Clearly, when $R \leq 1,000$ both our algorithm and PREMA are performing stably well, i.e., the NDE is no more than 5.2% while our algorithm has very little NDE improvement. When the rank is large, PREMA's performance drops a lot. For example, the NDE of PREMA is 14.9 when $R = 1,500$ in Case # 3. But, our algorithm has very stable NDE even when R is high. Interestingly, when the rank is large enough it's better for our algorithm to choose a large μ, which implies that the graph regularizer plays a critical role for the data disaggregation task. It's also worth to mention that from the Case #1 to Case #2 to Case #3, the compression rate is getting higher and higher and PREMA is facing the disaggregation challenge with a smaller and smaller rank. This implies that when the compression rate is low, both PREMA and our algorithm have prominent performance in a wide range of ranks. Last but not the least, our method is performing well in the extreme cases when either the rank is too high or the compression rate is too low (here, we aren't assuming the rank can be infinity high).

5 CONCLUSION

In this paper we introduce a graph-assisted tensor disaggregation method which leverages graph information to improve the robustness and the fidelity of the reconstruction of a high-resolution tensor from two complementary disaggregated views. In our experimental evaluation, we observe that our proposed method is able to operate well in cases where the rank of the underlying decomposition model is very high, where state of the art runs into instabilities