Efficient Tensor Decomposition Methods

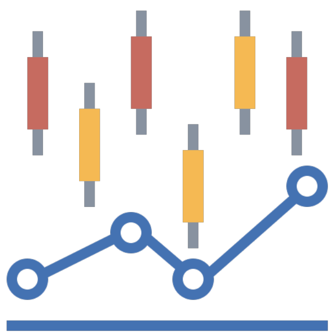
Jun-Gi Jang
Dept. of CS, UIUC

Outline

- **Introduction**
- Research Overview
- Efficient Method in Irregular Tensors
- Efficient Method for Diverse Time Ranges
- Conclusion

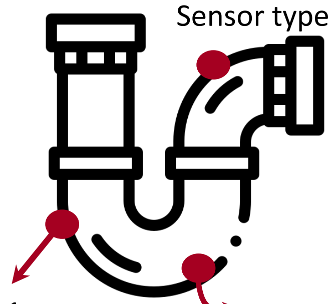
Tensors

- **Tensors** are everywhere
  - Tensors are a **generalization** of vectors (1-order tensors) and matrices (2-order tensors)
  - Example: a 3-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$



Stocks

3-order tensor
Index: (stock, feature, time)
Value: measurement



Sensor

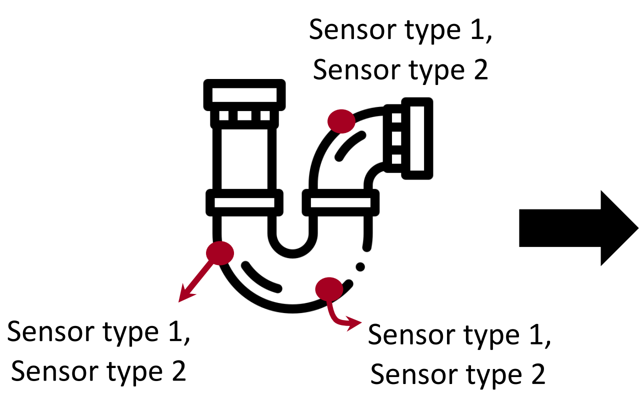
3-order tensor
Index: (sensor, location, time)
Value: measurement

Healthcare

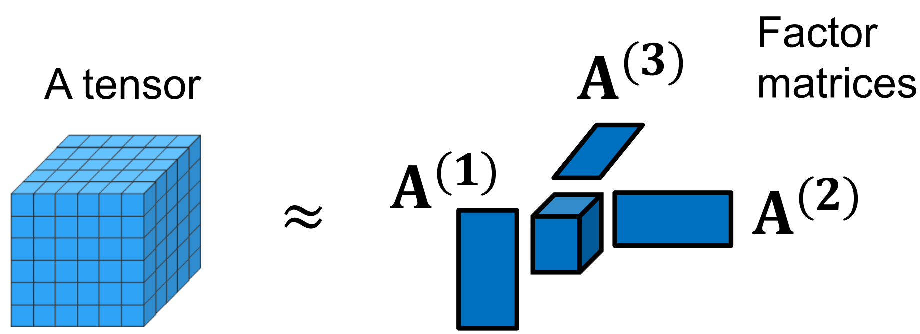
3-order tensor
Index: (patient, date, diagnosis)
Value: binary

Tensor Decomposition

- A fundamental tool to analyze tensors
- Find latent factor matrices which approximate an input tensor well
  - Find good representation vectors for all indices
    - For example, given a tensor of the size $2 \times 3 \times 4$, we need to find 9 latent factor vectors for 9 indices



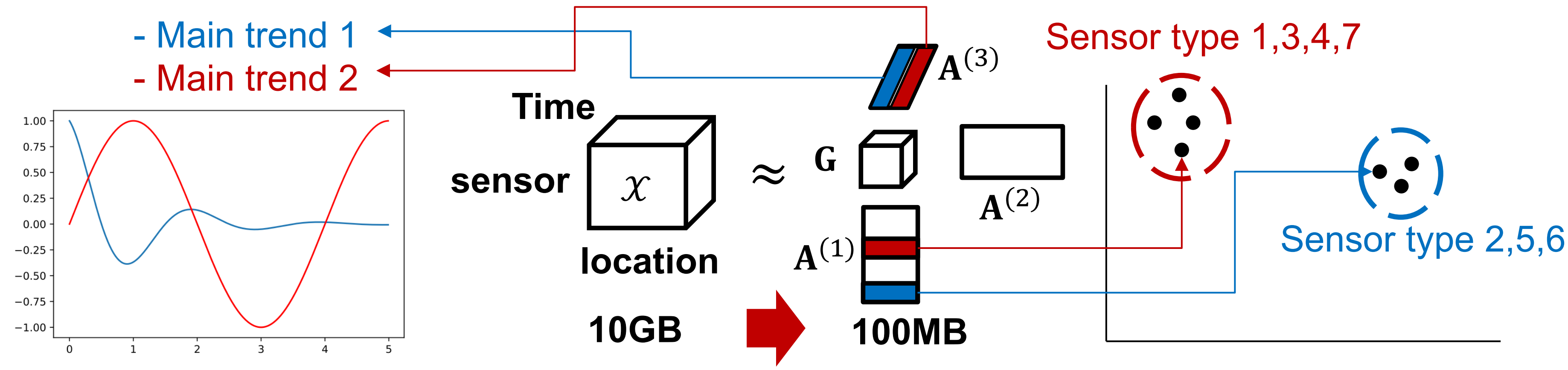
Sensor data



Example) Tucker decomposition

Applications

- **Several applications for Tensor decomposition**
  - Dimensionality reduction, concept discovery, trend analysis, anomaly detection, and clustering



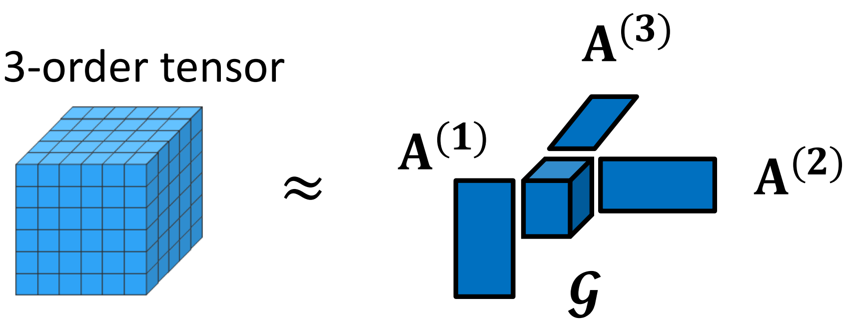
Trend Analysis
Find main trends using latent factors of the time dimension

Dimensionality Reduction
Lossy compression for a tensor

Clustering
Find similar objects using latent factors

Tucker Decomposition

- **Given** an N-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, target rank $J_1, \ldots, J_N$
- **Obtain** factor matrices $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times J}$ for $n = 1, \ldots, N$ and core tensor $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{J_1 \times \cdots \times J_N}$



Objective function

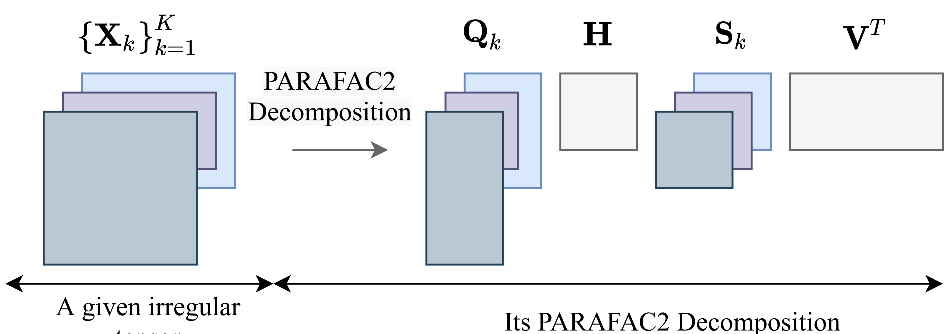
$$\min_{\boldsymbol{\mathcal{G}}, \{\mathbf{A}^{(k)}\}_{k=1}^{N}} \left\| \boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{G}} \times_{\mathbf{1}} \mathbf{A}^{(1)} \cdots \times_N \mathbf{A}^{(N)} \right\|_F^2$$

$\times_{\boldsymbol{n}}$**: n-mode product**

- The factor matrices and core tensor are used for real-world applications

PARAFAC2 Decomposition

- **Given** an irregular tensor $\{\mathbf{X}_k\}_{k=1}^{K}$, target rank R
  - Irregular tensor - a collection of matrices where the number of columns is the same and the number of rows is different from each other
- **Obtain** factor matrices $\mathbf{Q}_k \in \mathbb{R}^{I_k \times R}, \mathbf{H} \in \mathbb{R}^{R \times R}$, $\mathbf{S}_k \in \mathbb{R}^{R \times R}, \mathbf{V} \in \mathbb{R}^{J \times R}$ for k=1,…,K



Objective function

$$\min_{\mathbf{Q}_k,\mathbf{H},\mathbf{S}_k,\mathbf{V}} \sum_{k=1}^{K} \|\mathbf{X}_k - \mathbf{Q}_k \mathbf{H} \mathbf{S}_k \mathbf{V}^T\|_F^2$$

$\mathbf{Q}_k$: column orthogonal matrix
$\mathbf{S}_k$: diagonal matrix

- PARAFAC2 decomposition is tailored for irregular tensors

Alternating Least Square

- **ALS (Alternating Least Square)**
  - Find factor matrices of tensor decomposition including Tucker decomposition and PARAFAC2 decomposition
  - *Iteratively* updates a factor matrix while fixing the other factor matrices
    - Example) update $\mathbf{A}^{(1)}$ while fixing $\mathbf{A}^{(2)}$ and $\mathbf{A}^{(3)}$
  - Require computations with a given tensor at each iteration
    - Multiplication between factor matrices and an input tensor

Outline

- Introduction
- <u>**Research Overview**</u>
- Efficient Method in Irregular Tensors
- Efficient Method for Diverse Time Ranges
- Conclusion

Tensor decomposition Research

- Categorize tensor decomposition research
  - Data sparsity: Dense tensor/Sparse tensor
  - Metric: Efficiency/Accuracy
  - Decomposition method: Tucker/PARAFAC2/CP/Tensor Train, and so on…
  - Irregularity: Regular tensor/Irregular tensor
  - Setting: Static setting/Streaming setting/Time range setting (a custom setting)
- Can conduct research by selecting one keyword from each category

Research Goal

- **Research goal during my Ph.D. study is to develop efficient tensor decomposition-based methods in real-world settings**
  - Data sparsity: **Dense tensor**/Sparse tensor
  - Metric: **Efficiency**/Accuracy
  - Decomposition method: **Tucker/PARAFAC2**/CP/Tensor Train, and so on…
  - Irregularity: **Regular tensor/Irregular tensor**
  - Setting: **Static setting/Streaming setting/Time range setting (a custom setting)**

Research Outcome

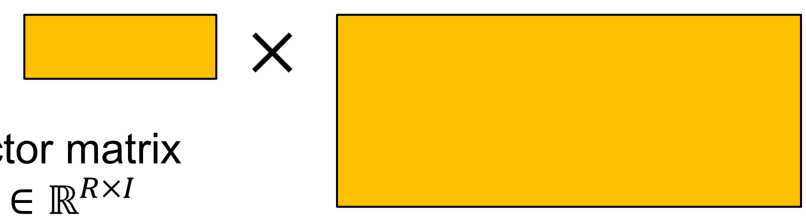
- **Improve the efficiency of tensor decomposition**

	Static Setting	Streaming Setting	Time Range Setting
Regular Dense Tensor	D-Tucker [ICDE20]	D-TuckerO [TKDD22]	**Zoom-Tucker [KDD21]**
Irregular Dense Tensor	**DPar2 [ICDE22]**	Dash [KDD23]	Developing …

Challenge

- To obtain factor matrices, we perform computations involved with an input tensor several times

Key operation in Tucker-ALS	Key operation in PARAFAC2-ALS
$\boldsymbol{\mathcal{X}} \times_1 \mathbf{A}^{(1)T} \cdots \times_N \mathbf{A}^{(N)T}$	$\mathbf{X}_k \mathbf{V} \mathbf{S}_k \mathbf{H}^T$ for all k

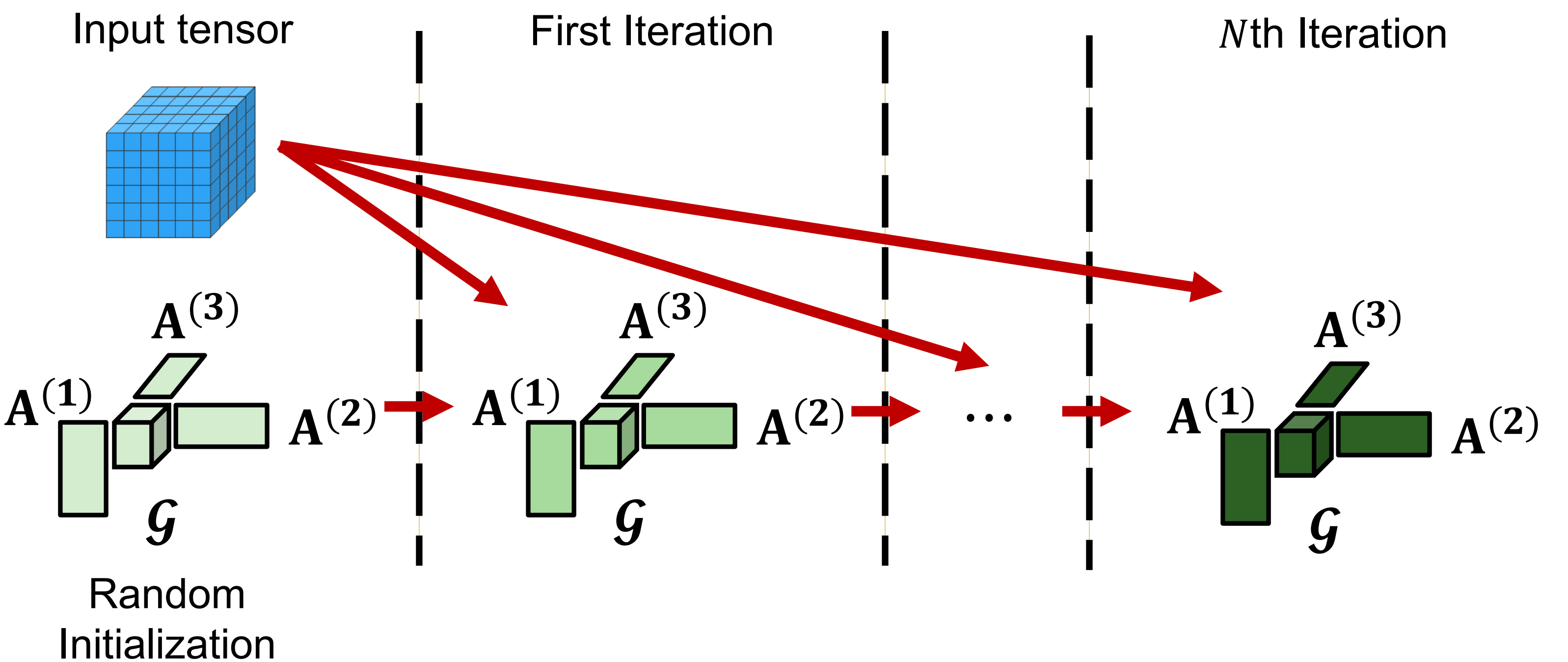


A factor matrix
$\mathbf{A} \in \mathbb{R}^{R \times I}$

A matrix reshaped from a tensor
$\mathbf{X} \in \mathbb{R}^{I \times (J \times K)}$

Challenge in Static Setting

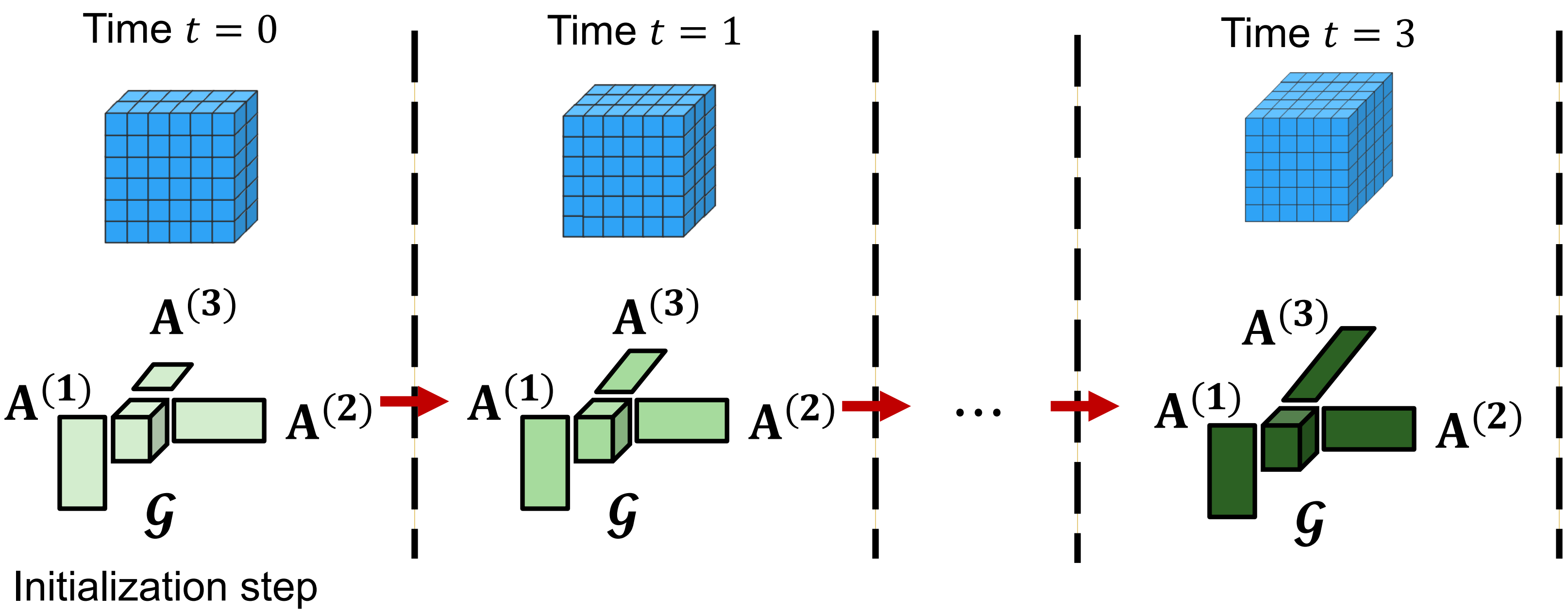
- In tensor decomposition, **iterative** computations with a given tensor require high costs



At each iteration, the computational cost is proportional to the size T of the input tensor $\Rightarrow$ require $O(NT)$ for the final outputs

Challenge in Streaming Setting

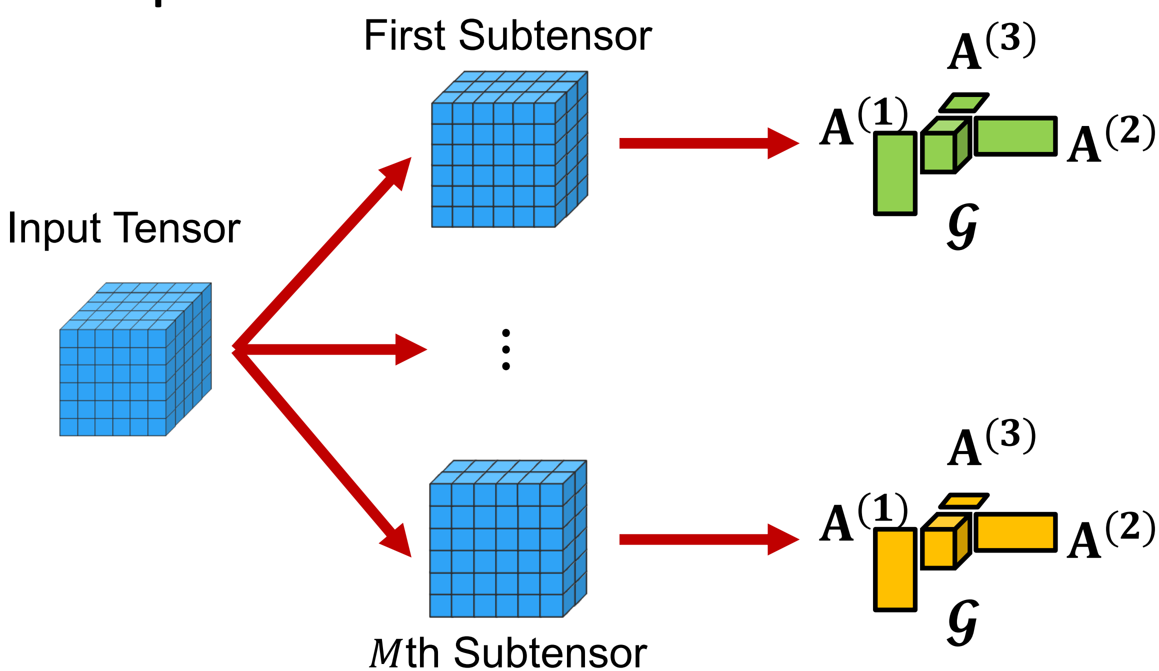
- Computations with the entire tensor require heavy cost as the size of the entire tensor increases over time



The computational cost of a naïve method is proportional to the size T of the entire tensor ⇒ require $O(T)$ at each time step t

Challenge in Time Range Setting

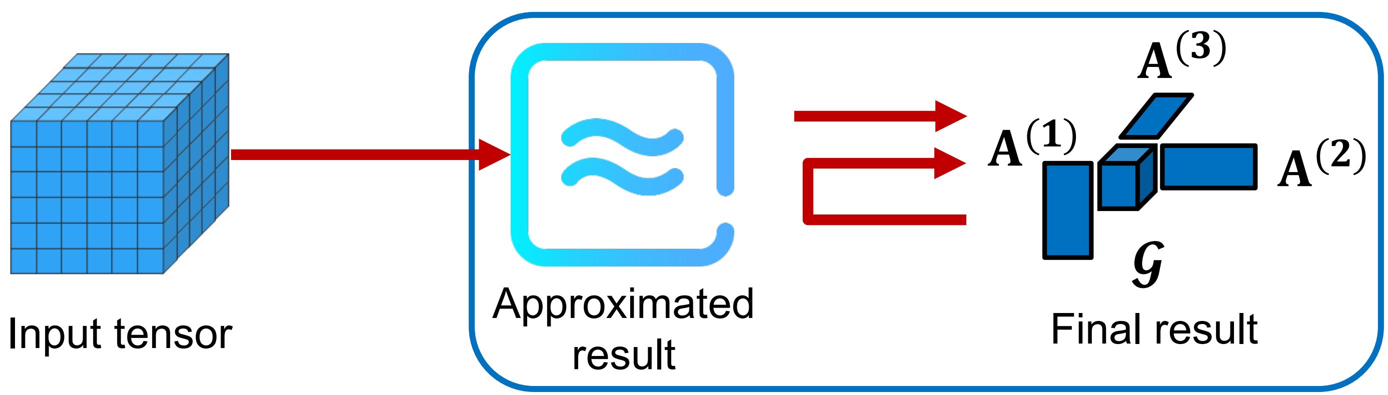
- Given a real-world tensor, performing tensor decomposition several times is also burdensome



The computational cost is proportional to the number M of subtensors whose sizes are approximately $S \Rightarrow$ require $O(NMS)$ for the final outputs

Main Technique

- Avoid repeated computations involved with a given tensor thoroughly
  1. Find approximated results
     - Obtained with high efficiency
     - Much smaller than an input tensor
  2. In repeated computations, exploit the approximated results to obtain the final results of tensor decomposition



Iterative computations
Cost: $O(T + NA) \ll O(NT)$
- N is the number of iterations
- A is the size of the approximation result
- T is the size of the input tensor

Outline

- Introduction
- Research Overview
- **Efficient Method in Irregular Tensors**
- Efficient Method for Diverse Time Ranges
- Conclusion

Research Outcome

- **Improve the efficiency of tensor decomposition**

	Static Setting	Streaming Setting	Time Range Setting
Regular Dense Tensor	D-Tucker [ICDE20]	D-TuckerO [TKDD22]	Zoom-Tucker [KDD21]
Irregular Dense Tensor	DPar2 [ICDE22]	Dash [KDD23]	Developing …

DPar2: Fast and Scalable PARAFAC2 Decomposition for Irregular Dense Tensors

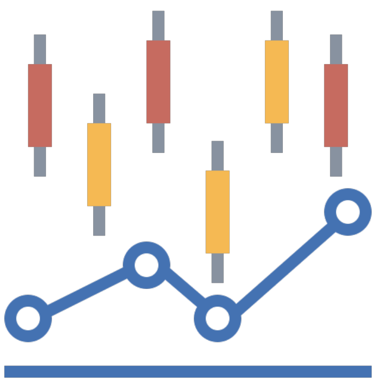
Best Paper
Honorable Mention

ICDE 2022

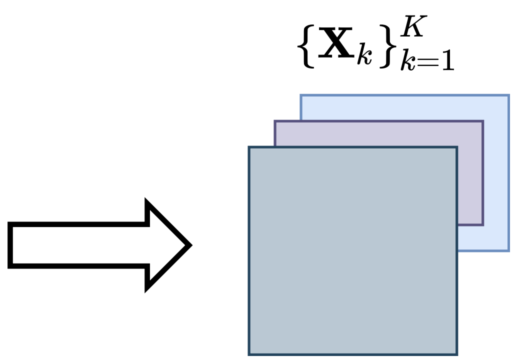
Jun-Gi Jang and U Kang
Data Mining Lab
Dept. of CSE
Seoul National University

Irregular Tensors

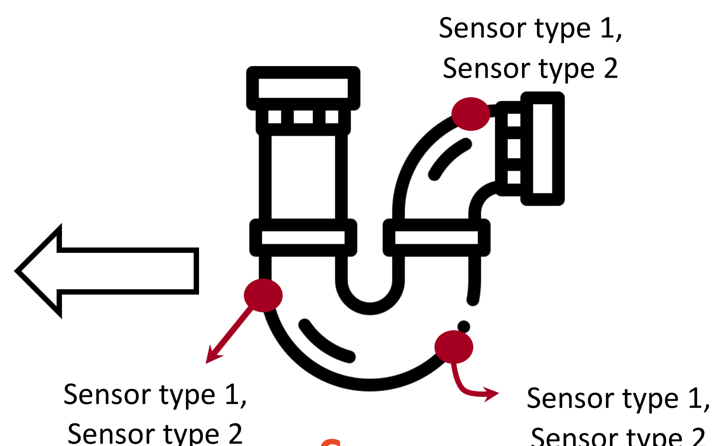
- Many real-world data are represented as **irregular tensors**
  - A collection of matrices where the number of columns is the same and the number of rows is different from each other



Stocks
Index: (time, feature, stock)
Value: measurement



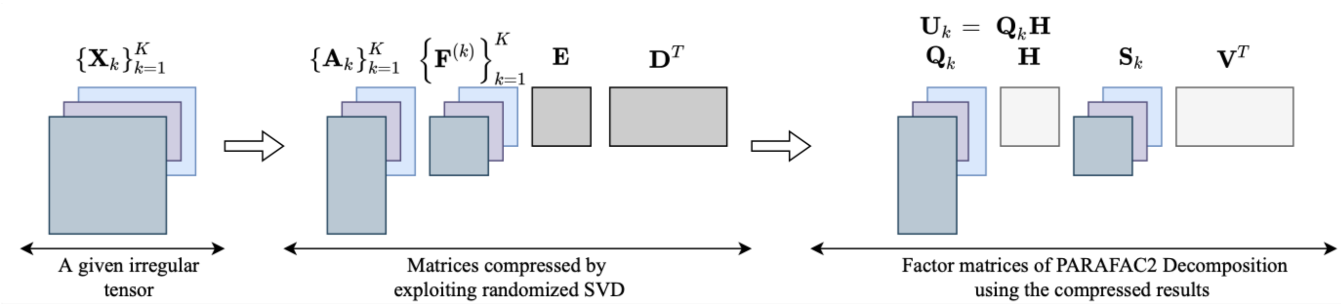
An irregular dense tensor



Sensor
Index: (time, location, sensor)
Value: measurement

Proposed Method

- We propose **DPar2** (**D**ense **PAR**AFAC**2** Decomposition)
  - A **fast** and **scalable** PARAFAC2 decomposition method for irregular dense tensors



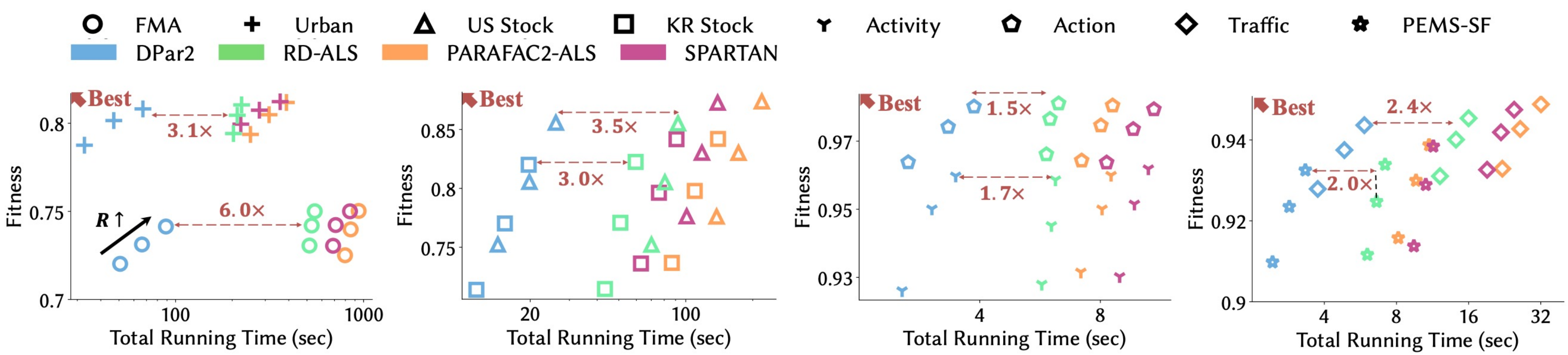
  - **(Idea 1)** Approximating an irregular tensor using randomized SVD (Singular Value Decomposition)
  - **(Idea 2)** Careful reordering of computations with the approximated results
    - Exploiting properties of operations and matrices
      - Toy example) $\mathbf{XV} \approx \mathbf{Y}(\mathbf{Z}^T\mathbf{V})$

$$O(IJR) \qquad O(IR^2 + JR^2)$$

$\mathbf{X} \in \mathbb{R}^{I \times J}$
$\mathbf{V} \in \mathbb{R}^{J \times R}$
$\mathbf{Y} \in \mathbb{R}^{I \times R}$
$\mathbf{Z} \in \mathbb{R}^{J \times R}$
$I, J \gg R$

Experiments
Trade-off

- DPar2 **outperforms** the competitors, giving up to **6× faster** than competitors while having comparable fitness
  - The upper-left region indicates better performance



(a) Trade-off (b) Trade-off (c) Trade-off (d) Trade-off

Outline

- Introduction
- Research Overview
- Efficient Method in Irregular Tensors
- **Efficient Method for Diverse Time Ranges**
- Conclusion

Research Outcome

- **Improve the efficiency of tensor decomposition**

	Static Setting	Streaming Setting	Time Range Setting
Regular Dense Tensor	D-Tucker [ICDE20]	D-TuckerO [TKDD22]	Zoom-Tucker [KDD21]
Irregular Dense Tensor	DPar2 [ICDE22]	Dash [KDD23]	Developing …

Fast and Memory-Efficient Tucker Decomposition for Answering Diverse Time Range Queries

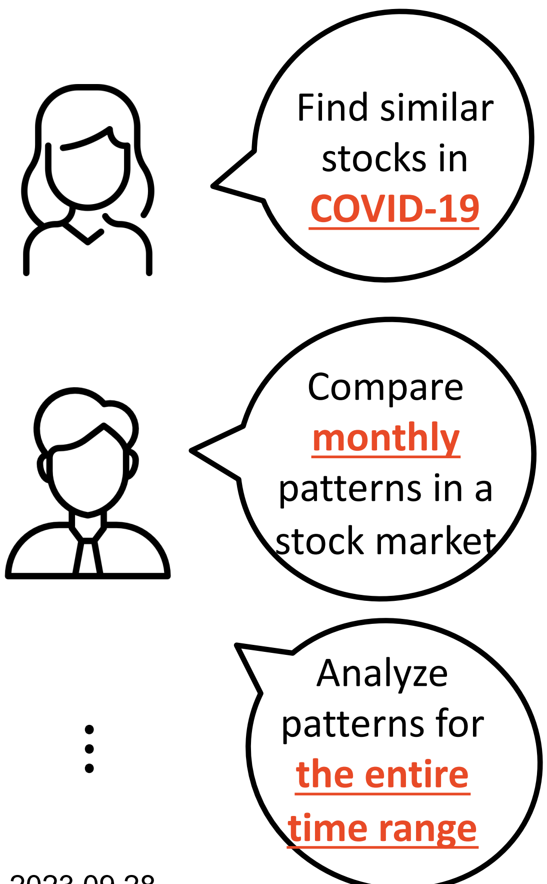
Best Research Paper

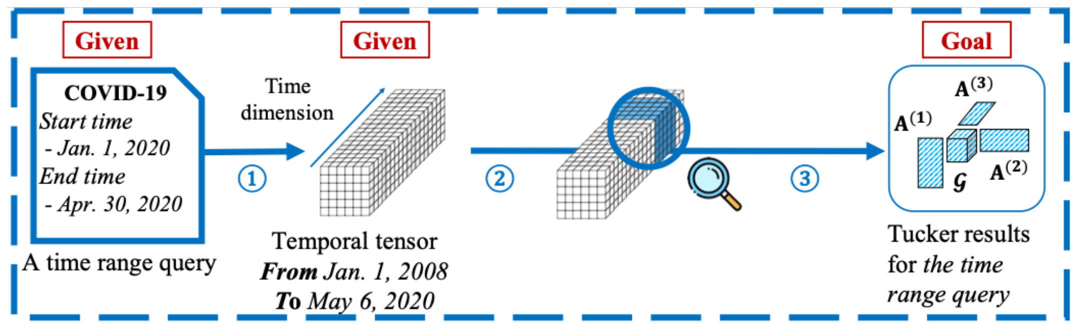
KDD 2021

Jun-Gi Jang and U Kang
Data Mining Lab
Dept. of CSE
Seoul National University

Research Problem
Time Range Problem

- Several users are interested in investigating patterns of **diverse** time ranges using Tucker decomposition





- **Given**
  - A temporal dense tensor $\mathcal{X}$
    - Assume the last dimension of the tensor is the time dimension
  - A time range query
- **Find:** the Tucker results of the sub-tensor of $\mathcal{X}$ included in the given range

Research Motivation

- **Limitations of previous works on the time range problem**
  - Previous methods focus on performing Tucker decomposition of **only the whole tensor once**
    - Example) Given a tensor and 1000 different time range queries, they need to perform Tucker decomposition 1000 times for different sub-tensors
  - **Require high time and space costs**
    - For various time ranges, there are plenty of redundant computations and a given tensor is large
- **Research Question**
  - How can we answer diverse time range queries quickly and memory-efficiently?

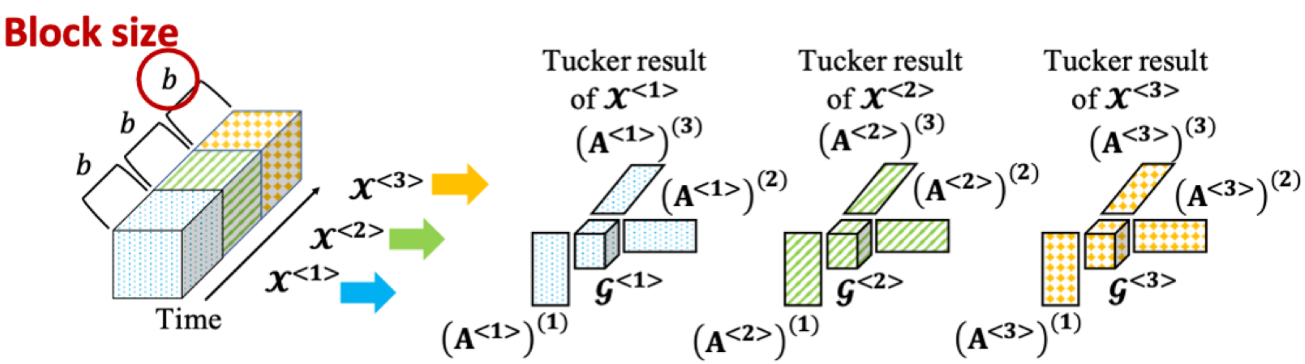
Proposed Method

- **Zoom-Tucker (Zoomable Tucker decomposition)**
  - Given a time range query, **efficiently** performs Tucker decomposition of the sub-tensor included in the time range
  - **The first method** that efficiently handles various time range queries in a tensor
- **Two phases**
  - **Preprocessing phase** effectively **approximates** a given temporal tensor block by block before time range queries are given
  - **Query phase** efficiently **answers** a given time range query by **exploiting** the approximated results

Preprocessing Phase

- Before time range queries are given, Zoom-Tucker approximates a given temporal tensor block by block along the time dimension
  - Split a given temporal tensor into sub-tensors along the time dimension

sition



$\mathcal{X}^{<i>}$: i-th block tensor
$(\mathbf{A}^{<i>})^{(n)}$: the n-th fact or matrix of i-th block te nsor
$\mathcal{G}^{<i>}$: the core tensor of i-th block tensor

- The advantages of the preprocessing phase
  1. Generate **small results** compared to an input tensor (support high efficiency of the query phase)
  2. Capture **local temporal information** (reduce error increase)

Query Phase

- Exploit mathematical techniques with the small approximated results for a given time range, achieving **high efficiency** for performing Tucker decomposition
  - Find factor matrices without reconstructing the approximated results
  - Minimize redundant computations in handling diverse time ranges
    - Carefully determine the order of computations

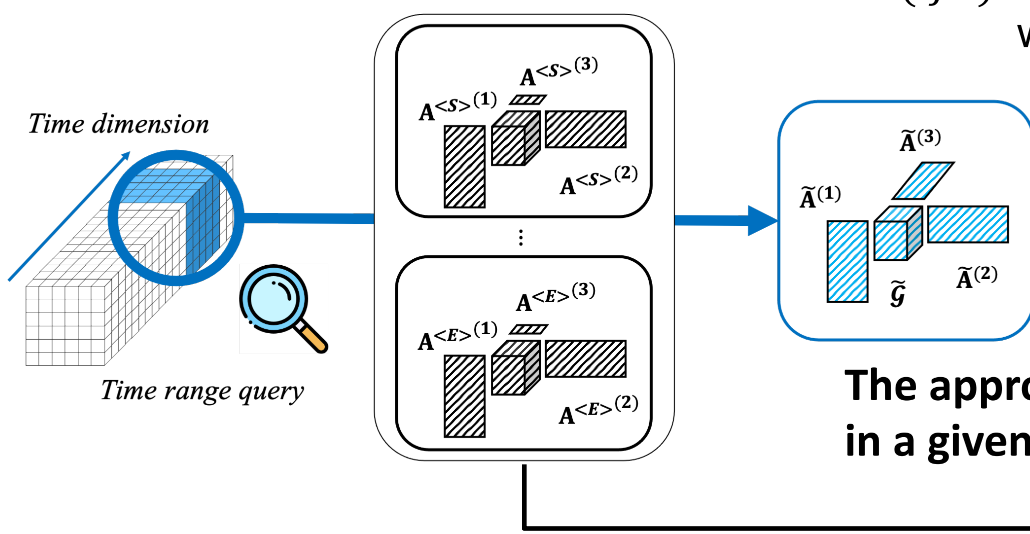
Toy example)

$O(IJR)$ $\quad \mathbf{A}^T\mathbf{X} \approx (\mathbf{A}^T\mathbf{Y})\mathbf{Z}^T$

where $\mathbf{X} \approx \mathbf{YZ}^T$

$\mathbf{A} \in \mathbb{R}^{I \times R}$
$\mathbf{X} \in \mathbb{R}^{I \times J}$
$\mathbf{Y} \in \mathbb{R}^{I \times R}$
$\mathbf{Z} \in \mathbb{R}^{J \times R}$
$I, J \gg R$

$O(IR^2 + JR^2)$

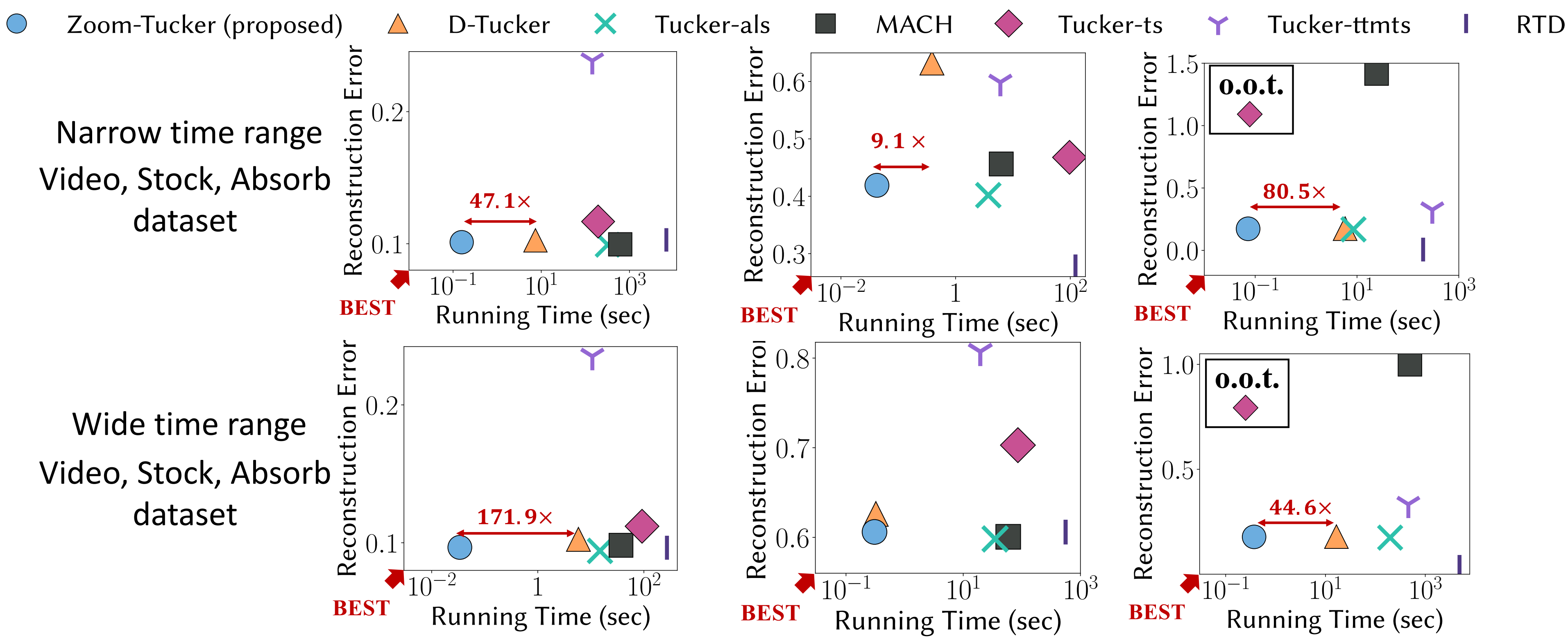


The Tucker result for the given query

The approximated results in a given time range query

Experiments
Trade-off

- Zoom-Tucker **outperforms** the competitors, giving up to **171× faster** than competitors while having comparable errors



Outline

- Introduction
- Research Overview
- Efficient Method in Irregular Tensors
- Efficient Method for Diverse Time Ranges
- **Conclusion**

Future Works

- Design a tensor method with deep learning techniques
  - Data sparsity: Dense tensor/**Sparse tensor**
  - Decomposition method: **Deep learning-based method**
  - Metric: Efficiency/**Accuracy**
  - Irregularity: **Regular tensor**/Irregular tensor
  - Setting: **Static setting**/Streaming setting/Time range setting (a custom setting)

Future Works

- There are a lot of research topics that can integrate with tensor decomposition
  - Many data can be represented as tensors
    - Time evolving graph
    - Knowledge graph
    - Multiple graphs
    - Temporal recommendation data
  - Develop a tensor-based method for real-world applications

Conclusion

- **Research Goals**
  - To design efficient tensor decomposition-based methods in real-world settings
    - **DPar2** is an efficient PARAFAC2 decomposition method for an irregular tensor
    - **Zoom-Tucker** is an efficient Tucker decomposition method for answering time range queries in a regular tensor
    - Other works improves the efficiency of tensor decomposition in various real-world settings
- **Future Plan**
  - Improving the accuracy of tensor decomposition
  - Design a tensor-based method for real-world applications

Thank you !