Network of Tensor Time Series

Presenter: Baoyu Jing
Contact: baoyuj2@Illinois.edu

Baoyu Jing
(UIUC)

Hanghang Tong
(UIUC)

Yada Zhu
(IBM)
Outline

• Introduction
• Preliminaries
• Methodology
• Experiments
• Conclusion
Outline

- Introduction
  - Preliminaries
  - Methodology
  - Experiments
  - Conclusion
Ubiquity of Co-evolving Time Series

- Co-evolving time series is ubiquitous.
  - Each time series is related to each other.

Environmental Monitoring  Financial Analysis  Smart Transportation
Properties of Co-evolving Time Series

- Take environmental monitoring as an example. (Left Figure)
- It is a tensor. (Middle Figure)
- Each temporal snapshot is a tensor. (The green slice)
- Network constraint for each dimension.

1. We have some monitoring sites/locations.
2. Each site has multiple types of sensors.
Challenge #1: Model Explicit Relations

• Network constraints.
  • Distance between the sensors.
  • Correlation between temperature, humidity, pressure etc.

✗ Existing methods are designed for flat graphs (e.g., GCN).
  • Either location network or type network, but not all.

✓ We introduce
  • Spectral Convolution for Tensor Graphs
  • Tensor Graph Convolutional Network (TGCN)

Challenge #2: Model Implicit Relations

- Different time series might have similar patterns.
  - E.g., air temperatures of Toronto and Mosco.
  - Explicit distance network constraint cannot capture this relation.

✗ Existing methods use
  - the same model for all time series
  - an individual model for each time series

✔ We introduce:
  - Tensor Recurrent Neural Network (TRNN)
  - Implement RNN with LSTM: TLSTM

Outline

• Introduction

➢ Preliminaries

• Methodology

• Experiments

• Conclusion
Preliminaries

• Tensor Graph $\mathcal{G}$:
  - A tensor graph is comprised of (1) a $M$-dimensional tensor $\mathcal{X} \in \mathbb{R}^{N_1 \times \cdots \times N_M}$ and (2) adjacency matrices $A_m \in \mathbb{R}^{N_m \times N_m}$.

• Network of Tensor Time Series:
  - It is comprised of a (1) tensor time series $\mathcal{S} \in \mathbb{R}^{N_1 \times \cdots \times N_M \times T}$, and (2) adjacency matrices $A_m \in \mathbb{R}^{N_m \times N_m}$.

• Mode-$m$ product between tensor $\mathcal{X}$ and matrix $U$: $\mathcal{X} \times_m U$
  - Generalization of the product between matrices.
Outline

• Introduction
• Preliminaries
  ➢ Methodology
• Experiments
• Conclusion
Overview

• Problem: given the $\omega$ historical snapshots, predict the next $\tau$ snapshots.
• Challenge 1: Explicit Relations - TGCN
• Challenge 2: Implicit Relations - TRNN
Tensor Graph Convolutional Network

\[ \tilde{\mathbf{x}} = \Phi^T \mathbf{x} \]

1. Graph Fourier Transformation -> Tensor Graph Fourier Transformation

\[ \tilde{\mathbf{x}} = \Phi^T \mathbf{x} \quad \Rightarrow \quad \tilde{\mathbf{x}} = \Phi^T_1 \mathbf{x} \]

2. Spectral Convolution

\[ \mathcal{G} \star \mathcal{X} = \mathcal{X} \prod_{m=1}^{M} \times_m \Phi_m^T \text{diag}(\tilde{g}_m) \Phi_m \]

3. Approximation via Chebyshev Polynomials

\[ \mathcal{G}_\theta \star \mathcal{X} = \mathcal{X} \prod_{m=1}^{M} \times_m \tilde{A}_m + \theta_0, \ldots, \theta M \]

4. Simplification

\[ \mathcal{G}_\theta \star \mathcal{X} = \sum_{\exists p_m=1}^{p_m=1} \mathcal{X} \prod_{m=1}^{M} \times_m \tilde{A}_m + \theta_0, \ldots \]

5. Updating function for each layer

\[ \text{TGCL}(\mathcal{X}, \{\mathcal{A}_m\}_{m=1}^{M}) = \sigma( \sum_{\exists p_m=1}^{p_m=1} \mathcal{X} \prod_{m=1}^{M} \times_m \tilde{A}_m \times_{M+1} \Theta_{p_1, \ldots, p_M} + \mathcal{X} \times_{M+1} \Theta_0) \]
TGCN: Detailed Analysis

• Let $m = 2$:

$$G_\theta \star X = \theta_{1,1}X \times_1 \tilde{A}_1 \times_2 \tilde{A}_2 + \theta_{1,0}X \times_1 \tilde{A}_1 + \theta_{0,1}X \times_2 \tilde{A}_2 + \theta_{0,0}X$$

Synergy of $A_1$ and $A_2$  Only $A_1$  Only $A_2$  Self-convolution or Residue connection

• Capture the synergy.
• Capture each network separately.
• Have a self-convolution/residue connection.
• Illustration of synergy:
  • node $v$ could gather information from $w'$.
Tensor Recurrent Neural Network

Tucker decomposition $U_m$ is orthonormal

$$Z_t = H_t \prod_{m=1}^{M} x_m U_m^T$$

Replace linear operations in RNN by multi-linear operations

$$\text{Linear}(x) = xw + b$$

$$\text{TLL}(X) = X \prod_{m=1}^{M+1} x_m W_m + b$$

Reuse $U_m$ since it is orthonormal.

$$R_t = Y_t \prod_{m=1}^{M} x_m U_m$$

We use LSTM to implement RNN.
The Implicit Relationship

• The Tucker decomposition is a high-order PCA or SVD.
  • $U_m$ extracts the eigenvectors of the $m$-th dimension.
  • Each element in $\mathcal{Z}$ indicates the interaction of the eigenvectors.
    • The degree of implicit relation.

• Let $\rho$ be the interaction degree: $N'_m = \rho N_m$ ($\forall m \in [1, \cdots, M]$)
  • $\rho \in (0, 1)$: ideal range
  • $\rho > 1$: $U_m$ is over-complete and have redundant information
  • $\rho = 0$: no interaction
Parameter Reduction

• Parameter Comparison: We use LSTM to implement TRNN -> TLSTM
  • TLSTM cell < multiple LSTM cells
  • Tucker decomposition introduces new parameters $U_m$
• TLSTM uses less parameters than multiple LSTM if:

$$
\text{Lemma 3.5 (Upper-bound for } \rho \text{). Let } N_m \text{ and } N'_m \text{ be the dimensions of } U_m \text{ in Equation (24), and let } d \in \mathbb{R} \text{ and } d' \in \mathbb{R} \text{ be the hidden dimensions of the inputs and outputs of TLSTM. TLSTM uses less parameters than multiple separate LSTMs, as long as the following condition holds:}
$$

$$
\rho \leq \sqrt{\frac{\left(\prod_{m=1}^{M} N_m - 1\right) d' (d + d' + 1)}{2 \sum_{m=1}^{M} N_m^2}} + \frac{1}{256} - \sqrt{\frac{1}{256}} \quad (33)
$$

$\rho$ be the interaction degree: $N'_m = \rho N_m \ (\forall m \in [1, \cdots, M])$

$d'$: the hidden dimension of LSTM/TLTSM

$N_m$: the dimension of the $m$-th mode of $\mathcal{H}_t$
Outline

• Introduction
• Preliminaries
• Methodology
  ➢ Experiments
• Conclusion
Experimental Setup

- Datasets:
  - Models
  - Soil
  - Revenue
- Tasks:
  - Missing Value Recovery
  - Future Value Prediction
- Metric: RMSE (the lower the better)
- Preprocessing:
  - Normalize each time series by z-scores.
  - Missing value recovery: use [0.1, 0.2, 0.3, 0.4, 0.5] for test
  - Future value prediction: use [0.02, 0.04, 0.06, 0.08, 0.1] for test
- Questions:
  - How accurate is NET$^3$ for missing value recovery and future value prediction?
  - How will synergy improve the performance?
  - How does the interaction degree $\rho$ impact the performance?
  - How efficient and scalable is NET$^3$?
Missing value recovery & Future value prediction

- The red arrows point (or the left-most bars) to NET$^3$.
- NET$^3$ performs the best: lowest RMSE.
Synergy Analysis

• Comparison methods:
  • (1) GCN with one network, (2) iTGCN: ~multiple GCNs, (3) TGCN: Full model

• TGCN (red arrows) performs the best.
Experiments: 20CR dataset

- The red arrows point (or the left-most bars) to the full model NET$^3$.
- NET$^3$ performs the best: lowest RMSE.
Visualization on the Traffic Dataset

- NET³ (red), Ground truth (black), Baselines (green, blue)
- NET³ performs the best: closest to the ground truth (see yellow circles).
Sensitivity

• As $\rho$ increases, the model performs better in general.
  • $U_m$ contains more eigenvectors: more information.
• # parameters of TLSTM grows linearly with $\rho$.

(a) Missing value recovery  
(b) Future value prediction  
(c) Number of parameters
Memory Efficiency

- **TLSTM** can significantly reduce # parameters.
- **TLSTM** achieves lower RMSE than **mLSTM**.

<table>
<thead>
<tr>
<th>Upper bound</th>
<th>Motes</th>
<th>Soil</th>
<th>Revenue</th>
<th>Traffic</th>
<th>20CR</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho_{\text{max}}$</td>
<td>2.17</td>
<td>2.43</td>
<td>0.64</td>
<td>0.31</td>
<td>57.25</td>
</tr>
<tr>
<td>$\rho_{\text{exp}}$</td>
<td>0.80</td>
<td>0.80</td>
<td>0.20</td>
<td>0.10</td>
<td>0.90</td>
</tr>
<tr>
<td><strong>TLSTM</strong></td>
<td>18,552</td>
<td>10,996</td>
<td>87,967</td>
<td>180,554</td>
<td>16,696</td>
</tr>
<tr>
<td><strong>mLSTM</strong></td>
<td>117,504</td>
<td>57,120</td>
<td>669,120</td>
<td>1,088,000</td>
<td>58,752,000</td>
</tr>
<tr>
<td><strong>Reduction ratio</strong></td>
<td>84.21%</td>
<td>80.75%</td>
<td>86.85%</td>
<td>83.40%</td>
<td>99.97%</td>
</tr>
</tbody>
</table>

(a) Motes-Missing  (e) Motes-Future
Scalability

• The training time V.S. size of input tensor: almost linear
• # parameters V.S. size of input tensor: almost linear.
Outline

• Introduction
• Preliminaries
• Methodology
• Experiments
➢ Conclusion
Conclusion

• Model Co-evolving time series
  • Challenge 1: Explicit Relationship
  • Solution 1: Tensor Graph Convolutional Network (TGCN)
  • Challenge 2: Implicit Relationship
  • Solution 2: Tensor Recurrent Neural Network (TRNN)

• Results:
  • NET$^3$ performs the best for missing value recovery and future value prediction.
  • TGCN captures the synergy among networks.
  • TRNN reduces # parameters and performs better.
Thank you!