Trajectory-User Linking using Higher-order Mobility Flow Representations
Supervisor: Manos Papagelis

M.SC. THESIS OF MAHMOUD ALSAEED
trajectories are unique to each person
Trajectory-user Linking (TUL)

trajectory-user linking aims to link anonymous trajectories to users who generate them
Data for Trajectory-user Linking (TUL)

Check-ins Trajectory

Mobility flow
Why do the limitations exist?

- Check-ins data
- Active data collection
Limitations of the current approaches

- Data Quality
  - low accuracy and completeness

- Data sparsity
  - limited data

- Imbalanced Data
  - 80% of the data is generated by 20% of the users
Problem Definition
What is a check-in trajectory?

- **Check-in record/visit**
  - \( r = (u, p, t, \langle x, y \rangle) \)

- **Check-in trajectory**
  - \( Tr = \{ r_1, r_2, \ldots, r_m \} \)
Problem Definition

Trajectory-user linking

- aims to link anonymous trajectories to users who generate them

\[ \mathcal{F} = \{ T_{r_1}, T_{r_2}, \ldots, T_{r_n} \} \] - trajectories

\[ \mathcal{U} = \{ u_1, u_2, u_3, \ldots, u_c \} \] - users

\[
\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(y_i - y'_i)
\]

\( \mathcal{F} \) is the set of all classifiers in the hypothesis space

\( \mathcal{L}(\cdot) \) is the loss between the predicted label \( y'_i \) and the true label \( y_i \).
Methodology
Overview

▷ Generating higher-order mobility flow representations
  • generating mobility flow data from check-ins
  • generating higher-order mobility flow and check-ins

▷ Modeling trajectory-User linking
Generating Mobility flow data

Check-ins Trajectory

Mobility flow
Mobility flow of NYC and TKY

NYC

TKY
Generating higher-order check-ins

Check-ins Trajectory

Higher-order check-ins
Translate check-ins to Higher-order

Check-ins

\[ Tr = \{ r_1, r_2, \ldots, r_m \} = \{(p_1, t_1, (x_1, y_1)), (p_2, t_2, (x_2, y_2)), \ldots, (p_m, t_m, (x_m, y_m))\} \]

Higher-order

\[ \{(p_1, t_1, g_1), (p_2, t_2, g_2), \ldots, (p_m, t_m, g_m)\} \]
Generating Higher-order Mobility flow

Mobility flow

Higher-order Mobility flow
Higher-order Mobility flow data

- Represented as a sequence of grid cells

- \( \{g_1, g_2, \ldots \} \)
FOURSQUARE-NYC Heatmap

Higher-order check-ins

Higher-order Mobility flow
How to calculate Sparsity?

Sparsity = % of zeros in User-POI matrix

\[
\frac{3}{9} = 30\%
\]
Higher-order Sparsity

<table>
<thead>
<tr>
<th></th>
<th>Alex</th>
<th>Eve</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td>{1,3}</td>
<td>{g_1, g_2}</td>
<td>{g_2}</td>
<td>{g_1, g_2, g_2}</td>
</tr>
</tbody>
</table>

\[ \begin{bmatrix} g_1 & g_2 \\ 1 & 1 \\ 0 & 1 \\ 1 & 2 \end{bmatrix} \]

\[ \frac{1}{6} = 16\% \]
Check-in Sparsity $\geq$ Higher-order Sparsity
Impact of higher-order abstraction on sparsity

- NYC
- TKY

Sparsity:
- 99.0%
- 98.0%
- 97.0%
- 96.0%
- 95.0%
- 94.0%
- 93.0%
- 92.0%

Types of data:
- Check-ins
- Higher-order check-ins
- Mobility flow
Overview

› Generating higher-order mobility flow representations
  • generating mobility flow data from check-ins
  • generating higher-order mobility flow and check-ins

› Modeling trajectory-User linking
TULHOR (trajectory-user linking using higher-order representations)
Spatial embedding

Higher-order Mobility flow

Spatial Embedding

Node2Vec

Mobility flow graph
TULHOR (trajectory-user linking using higher-order representations)
Spatial-temporal embedding

Higher-order check-ins \( \{(p_1, t_1, g_1), (p_2, t_2, g_2), \ldots, (p_m, t_m, g_m)\} \)

\[
\begin{align*}
\mathbf{z}_i^g &= \phi_g(g_i, W_g) \\
\mathbf{z}_i^p &= \phi_p(p_i, W_p) \\
\mathbf{z}_i^s &= \phi_s(g_i, W_s)
\end{align*}
\]
Temporal-aware positional encoding

\[ \{(p_1, t_1, g_1), (p_2, t_2, g_2), \ldots, (p_m, t_m, g_m)\} \]

\[ [z^t_i]_j = \begin{cases} 
\sin(w_j t_i), & \text{if } j \text{ is odd} \\
\cos(w_j t_i), & \text{if } j \text{ is even} 
\end{cases} \]
Output of the Spatial-temporal embedding

\[ R^{(id)} = z^g_1, z^g_2, \ldots, z^g_m \]

\[ R = (\{z^s_1, z^s_2, \ldots, z^s_m\}, \{z^p_1, z^p_2, \ldots, z^p_m\}, \{z^t_1, z^t_2, \ldots, z^t_m\}) \]
TULHOR

Encoder

Spatial-Temporal embedding

Masked token prediction

Feed Forward

ST-NOVA

Spatial embedding

Higher-order check-ins

Higher-order mobility flow

p_1 \ t_1 \ g_1 \ p_2 \ t_2 \ g_2 \ \ldots \ \ldots \ \ldots \ p_m \ t_m \ g_m
\[ {\text{ST-NOVA}}(R^{(id)}, R) = \sigma \left( \frac{QK^T}{\sqrt{d_L}} \right) V \]

\[ Q = R^{id} \times W_Q, \quad K = F \times W_k, \quad V = F \times W_V \]

\[ F = MLP(R^{(id)} \| R) \]
TULHOR

Encoder

Masked token prediction

Feed Forward

ST-NOVA

Spatial-Temporal embedding

Higher-order check-ins

Spatial embedding

Higher-order mobility flow

$z^g_1 \ z^g_2 \ z^g_3 \ mask \ \ldots \ \ z^g_m$

$z^t_1 \ z^t_2 \ z^t_3 \ mask \ \ldots \ \ z^t_m$

$z^p_1 \ z^p_2 \ z^p_3 \ mask \ \ldots \ \ z^p_m$

$z^s_1 \ z^s_2 \ z^s_3 \ mask \ \ldots \ \ z^s_m$

$p_1 \ t_1 \ g_1 \ p_2 \ t_2 \ g_2 \ \ldots \ p_m \ t_m \ g_m$
Two stages

- **Pre-training TULHOR**
  - Input: higher-order check-ins + Masking, higher-order mobility flow
  - Output: predicting masked token

- **Fine-tuning TULHOR**
  - Input: higher-order check-ins
  - Output: user who generated the higher-order check-ins
Experiment
Overview

Datasets
• Foursquare NYC and TKY

Experiments
• TULHOR accuracy performance (vs SOTA and baselines)
• TULHOR Ablation study
• Tessellation granularity (grid size) effect
| DATASET          | $|U|$  | $|T|$  |
|-----------------|-----|-----|
|                 | 108 | 6795|
| FOURSQUARE-NYC  | 209 | 9,637|
|                 | 234 | 10,133|
|                 | 108 | 9343|
| FOURSQUARE-TKY  | 209 | 14,151|
|                 | 451 | 20,964|
Baselines

Conventional ML:
- Decision Tree
- Linear Discriminant Analysis (LDA)
- Linear Support Vector Machine (SVM)

TULER:
- RNN
- LSTM
- GRU

DeepTUL
- RNN (DeepTUL)
- LSTM (Attn-LSTM)
- GRU (Attn-GRU)
## TULHOR performance

<table>
<thead>
<tr>
<th>Model</th>
<th>$\mathcal{U} = 108$</th>
<th>$\mathcal{U} = 209$</th>
<th>$\mathcal{U} = 451$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC@1</td>
<td>ACC@5</td>
<td>P</td>
</tr>
<tr>
<td>DT</td>
<td>0.789</td>
<td>0.793</td>
<td>0.785</td>
</tr>
<tr>
<td>LDA</td>
<td>0.853</td>
<td>0.912</td>
<td>0.927</td>
</tr>
<tr>
<td>LINEAR-SVM</td>
<td>0.890</td>
<td>0.948</td>
<td>0.923</td>
</tr>
<tr>
<td>TULER</td>
<td>0.870</td>
<td>0.933</td>
<td>0.871</td>
</tr>
<tr>
<td>TULER-L</td>
<td>0.905</td>
<td>0.952</td>
<td>0.904</td>
</tr>
<tr>
<td>TULER-G</td>
<td>0.915</td>
<td>0.954</td>
<td>0.916</td>
</tr>
<tr>
<td>ATT-LSTM</td>
<td>0.908</td>
<td>0.966</td>
<td>0.916</td>
</tr>
<tr>
<td>ATT-GRU</td>
<td>0.933</td>
<td>0.975</td>
<td>0.932</td>
</tr>
<tr>
<td>DeepTUL</td>
<td>0.922</td>
<td>0.966</td>
<td>0.927</td>
</tr>
<tr>
<td>TULHOR</td>
<td><strong>0.939</strong></td>
<td><strong>0.973</strong></td>
<td><strong>0.937</strong></td>
</tr>
<tr>
<td>Improvement</td>
<td>0.58%</td>
<td>-0.26%</td>
<td>0.59%</td>
</tr>
</tbody>
</table>

**TULHOR outperforms every baseline**

**TULHOR has better scalability**
| MODEL     | \(|\mathcal{U}| = 108\) | \(|\mathcal{U}| = 209\) | \(|\mathcal{U}| = 234\) |
|-----------|---------------------------|---------------------------|---------------------------|
|           | ACC@1 | ACC@5 | P   | R   | F1   | ACC@1 | ACC@5 | P   | R   | F1   | ACC@1 | ACC@5 | P   | R   | F1   |
| DT        | 0.884 | 0.892 | 0.878 | 0.867 | 0.868 | 0.785 | 0.788 | 0.753 | 0.728 | 0.730 | 0.778 | 0.782 | 0.722 | 0.712 | 0.705 |
| LDA       | 0.822 | 0.851 | 0.962 | 0.810 | 0.868 | 0.746 | 0.781 | 0.791 | 0.687 | 0.718 | 0.696 | 0.752 | 0.724 | 0.615 | 0.650 |
| LINEAR-SVM | 0.873 | 0.929 | 0.966 | 0.878 | 0.909 | 0.776 | 0.839 | 0.785 | 0.702 | 0.727 | 0.731 | 0.798 | 0.724 | 0.628 | 0.657 |
| TULER     | 0.870 | 0.929 | 0.869 | 0.851 | 0.852 | 0.776 | 0.853 | 0.749 | 0.722 | 0.718 | 0.768 | 0.844 | 0.733 | 0.707 | 0.703 |
| TULER-L   | 0.903 | 0.942 | 0.904 | 0.890 | 0.890 | 0.847 | 0.898 | 0.828 | 0.803 | 0.807 | 0.845 | 0.889 | 0.821 | 0.806 | 0.803 |
| TULER-G   | 0.909 | 0.949 | 0.914 | 0.897 | 0.898 | 0.854 | 0.892 | 0.853 | 0.811 | 0.812 | 0.846 | 0.891 | 0.821 | 0.805 | 0.803 |
| ATT-LSTM  | 0.823 | 0.896 | 0.715 | 0.703 | 0.709 | 0.716 | 0.832 | 0.554 | 0.559 | 0.556 | 0.712 | 0.830 | 0.569 | 0.557 | 0.563 |
| ATT-GRU   | 0.886 | 0.933 | 0.779 | 0.779 | 0.791 | 0.835 | 0.891 | 0.663 | 0.680 | 0.671 | 0.889 | 0.936 | 0.741 | 0.738 | 0.740 |
| DEEPTUL   | 0.853 | 0.923 | 0.765 | 0.738 | 0.751 | 0.733 | 0.840 | 0.614 | 0.597 | 0.606 | 0.789 | 0.891 | 0.607 | 0.617 | 0.612 |
| TULHOR    | 0.940 | 0.966 | 0.938 | 0.931 | 0.932 | 0.903 | 0.943 | 0.890 | 0.877 | 0.876 | 0.892 | 0.932 | 0.876 | 0.864 | 0.860 |
| Improvement | 3.42% | 1.85% | -2.89% | 3.85% | 2.53% | 5.82% | 5.07% | 6.58% | 7.83% | 7.87% | 0.35% | -0.49% | 6.61% | 7.13% | 7.19% |
Ablation study

Removing Higher-order significantly reduces the performance
## Tessellation granularity (grid size) effect

<table>
<thead>
<tr>
<th>Resolution</th>
<th># of Cells</th>
<th>Cell Size (km(^2))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex@7</td>
<td>334</td>
<td>5.160</td>
</tr>
<tr>
<td>Hex@8</td>
<td>2,003</td>
<td>0.730</td>
</tr>
<tr>
<td>Hex@9</td>
<td>11,036</td>
<td>0.015</td>
</tr>
</tbody>
</table>
Tessellations of Tokyo
# Results of grid size study

<table>
<thead>
<tr>
<th>Method</th>
<th># Users = 108</th>
<th>FOURSQUARE-TKY</th>
<th># Users = 209</th>
<th># Users = 451</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACC@1</td>
<td>ACC@5</td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>HEX@7</td>
<td>0.923</td>
<td>0.971</td>
<td>0.920</td>
<td>0.911</td>
</tr>
<tr>
<td>HEX@8</td>
<td>0.926</td>
<td>0.977</td>
<td>0.925</td>
<td>0.917</td>
</tr>
<tr>
<td>HEX@9</td>
<td>0.939</td>
<td>0.973</td>
<td>0.937</td>
<td>0.934</td>
</tr>
</tbody>
</table>

Hex@9 outperforms other sizes as the number of users increases.

The smaller the cells are, the better the scalability.
Computing Environment

- Implemented in Python
- Run in a GPU cluster (lion)
Conclusion
Summary

- Method for generating higher-order mobility flow & check-ins
- Novel deep learning framework – TULHOR (trajectory-user linking using higher-order representations)
- Extensive set of experiments
Contributions

- Model agnostic method for dealing with sparsity and low data quality
- Implementation of multiple TUL models
- Publicly available dataset
- Accepted to The 24th IEEE International Conference on Mobile Data Management (MDM)
Future works

- Multi-trajectory User Linking
- POI recommendation
- Mixed hierarchical representation
Impact of Sparsity

The graph shows the impact of sparsity on location accuracy. The x-axis represents different scenarios of check-ins and hexagons, and the y-axis represents sparsity levels. The accuracy is measured by F1 score. The graph indicates that as sparsity increases, the F1 score decreases, highlighting the importance of maintaining high sparsity levels for accurate location tracking.
Balanced loss

\[ \mathcal{L}(T_r, u_i) = \frac{1 - \beta}{1 - \beta n_u_i} \log(\sigma(y')) \]