wsdm
Phoenix, AZ, USA
2022
The 15th ACM International Conference on
Web Search and Data Mining
February 21–25, 2022

# S-Walk: Accurate and Scalable Session-based Recommendation with Random Walks

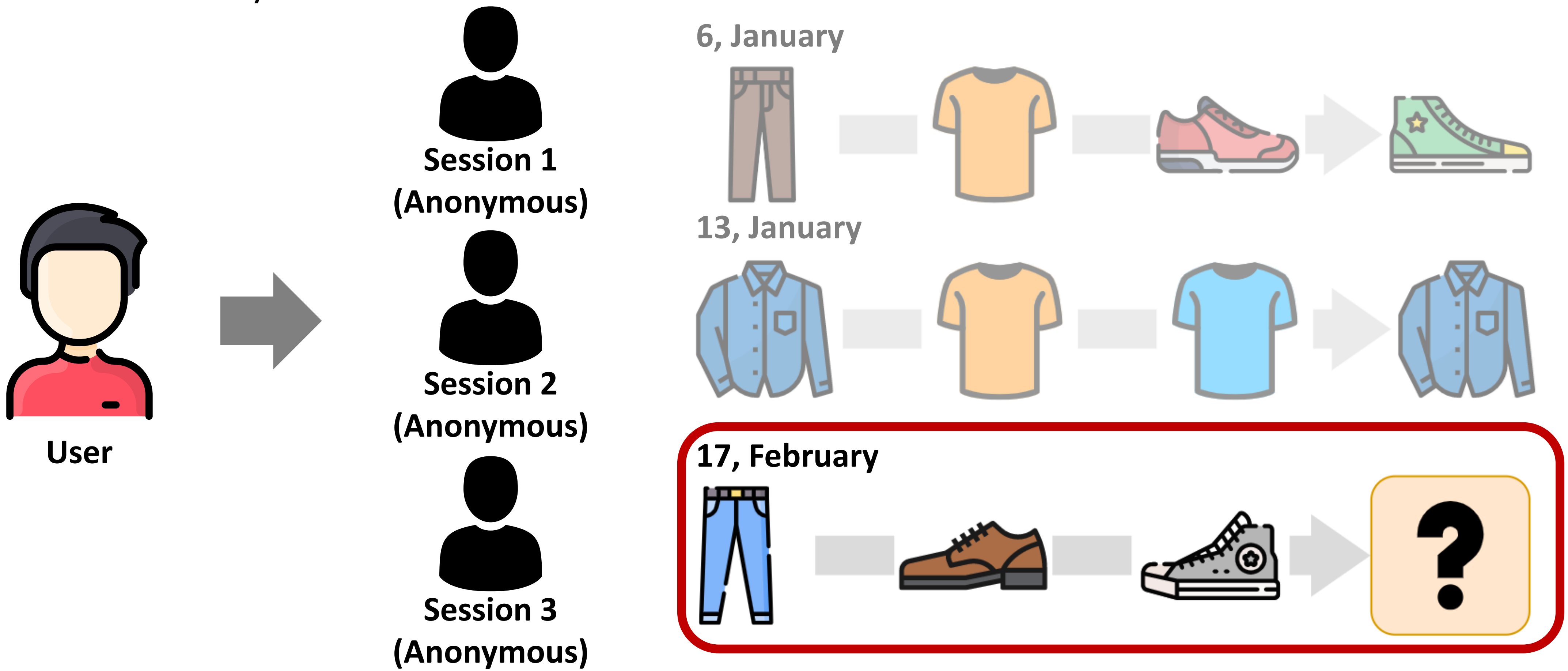
Minjin Choi[1], Jinhong Kim[1], Joonseok Lee[2,3] , Hyunjung Shim[4], Jongwuk Lee[1]

Sungkyunkwan University (SKKU)[1], Google Research[2]
Seoul National University[3], Yonsei University[4]

# Motivation

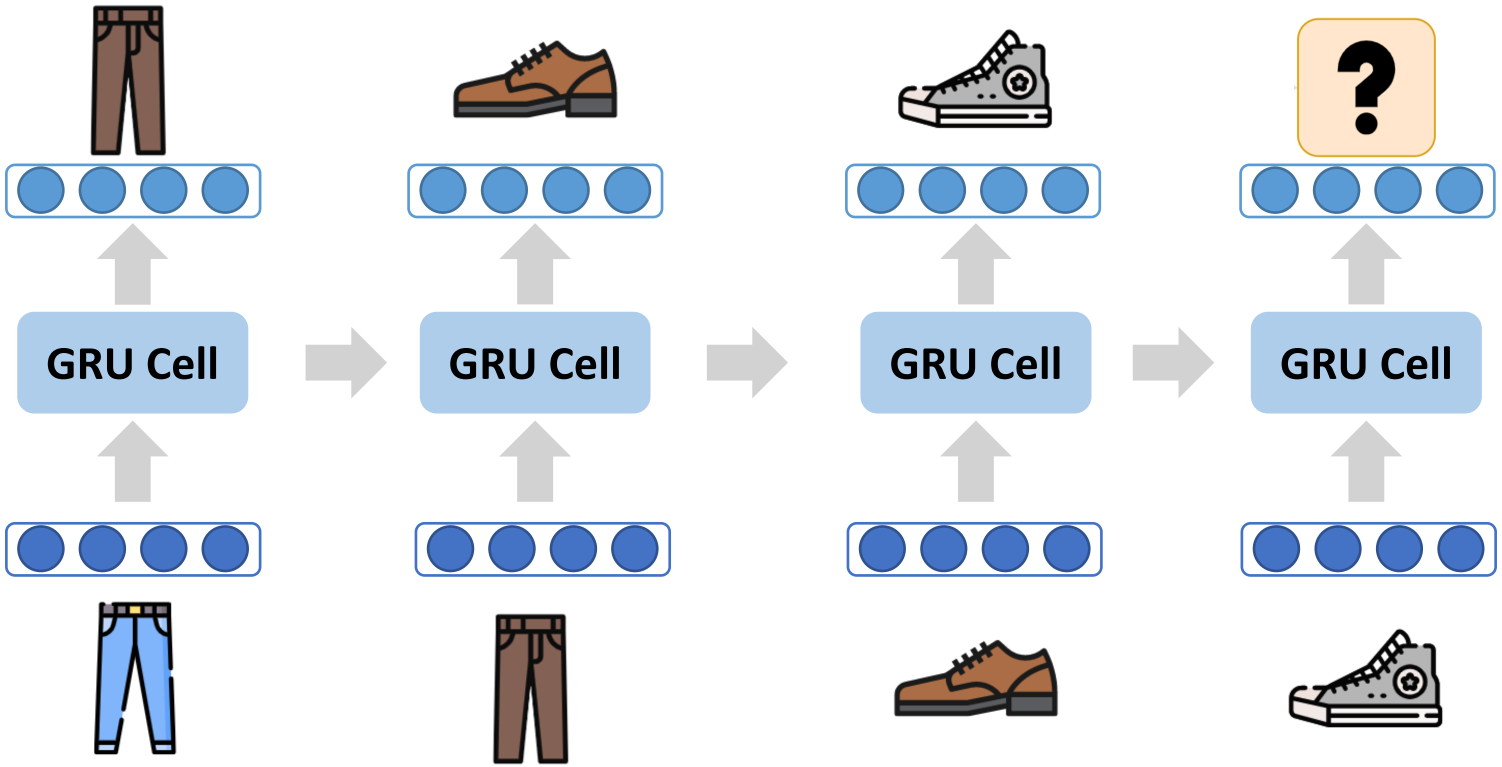
# Session-based Recommendation (SR)

- **Predicting the next item(s) based on only the current session**
  - Challenge: (1) The user ID cannot be used, (2) session histories are extremely short.



# Limitation of Existing SR Models

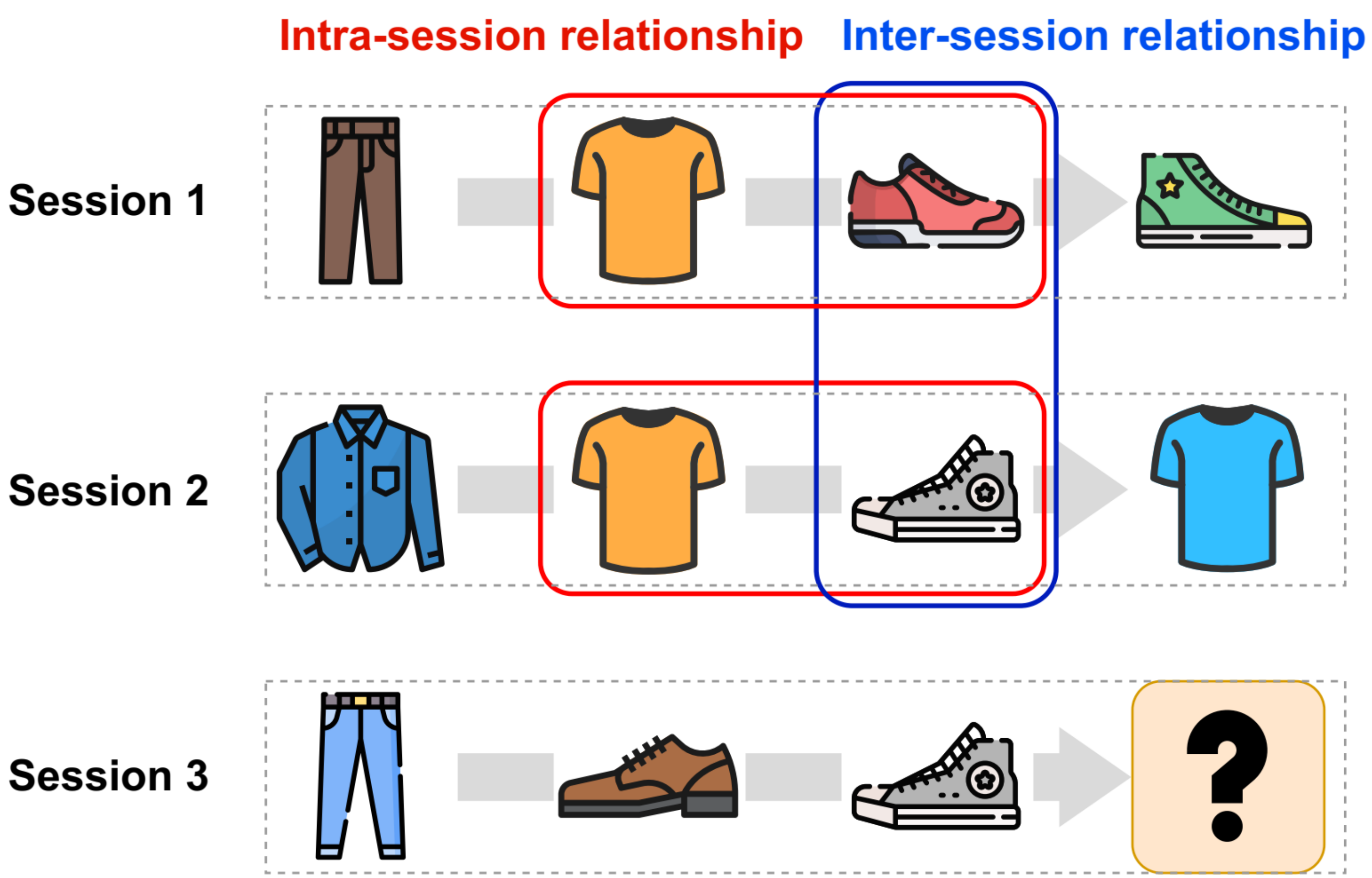
- **DNN models show good performance but suffer scalability issues.**
  - RNNs and GNNs are mostly used for session-based recommendations.
  - When the dataset is too large, the scalability issue arises.




*Balázs Hidasi et. al., "Session-based Recommendations with Recurrent Neural Networks", ICLR 2016*

# Limitation of Existing SR Models

➢ Most SR models pay less attention to inter-session relationships.

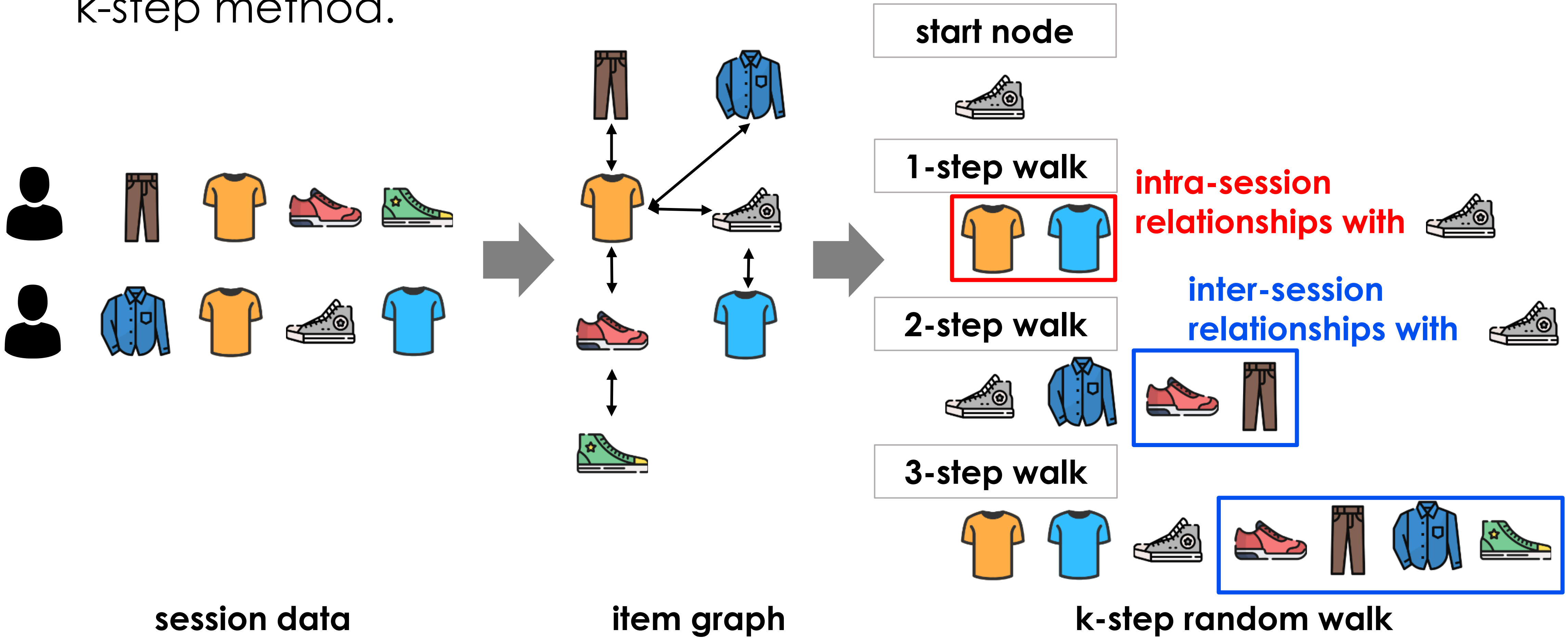


# Research Question

How to build the accurate and scalable session-based recommender model?

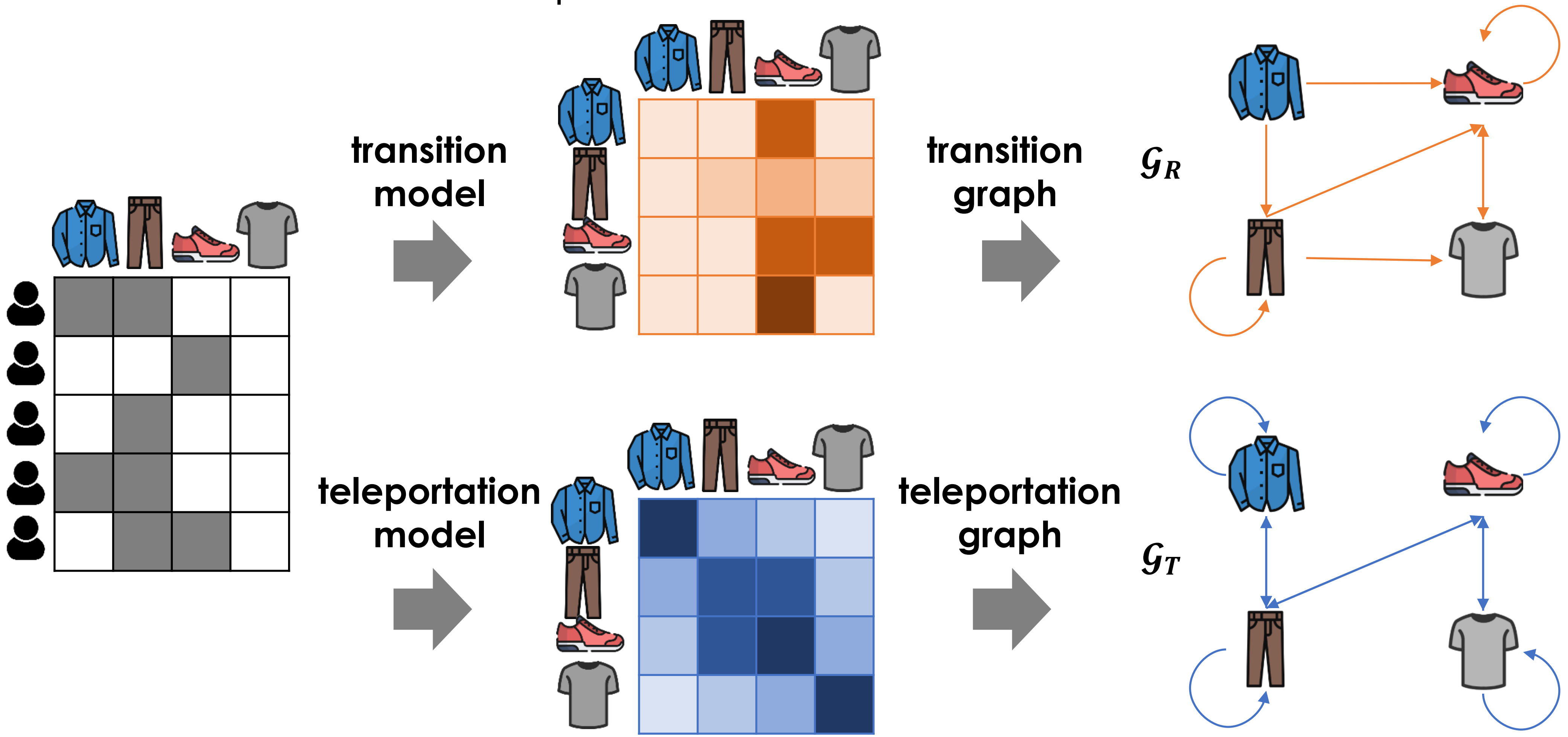
# Our Key Contributions

- **We utilize random walks to exploit session relationships.**
  - For the proposed model, we adopt random walk with restarts instead of k-step method.



# Our Key Contributions

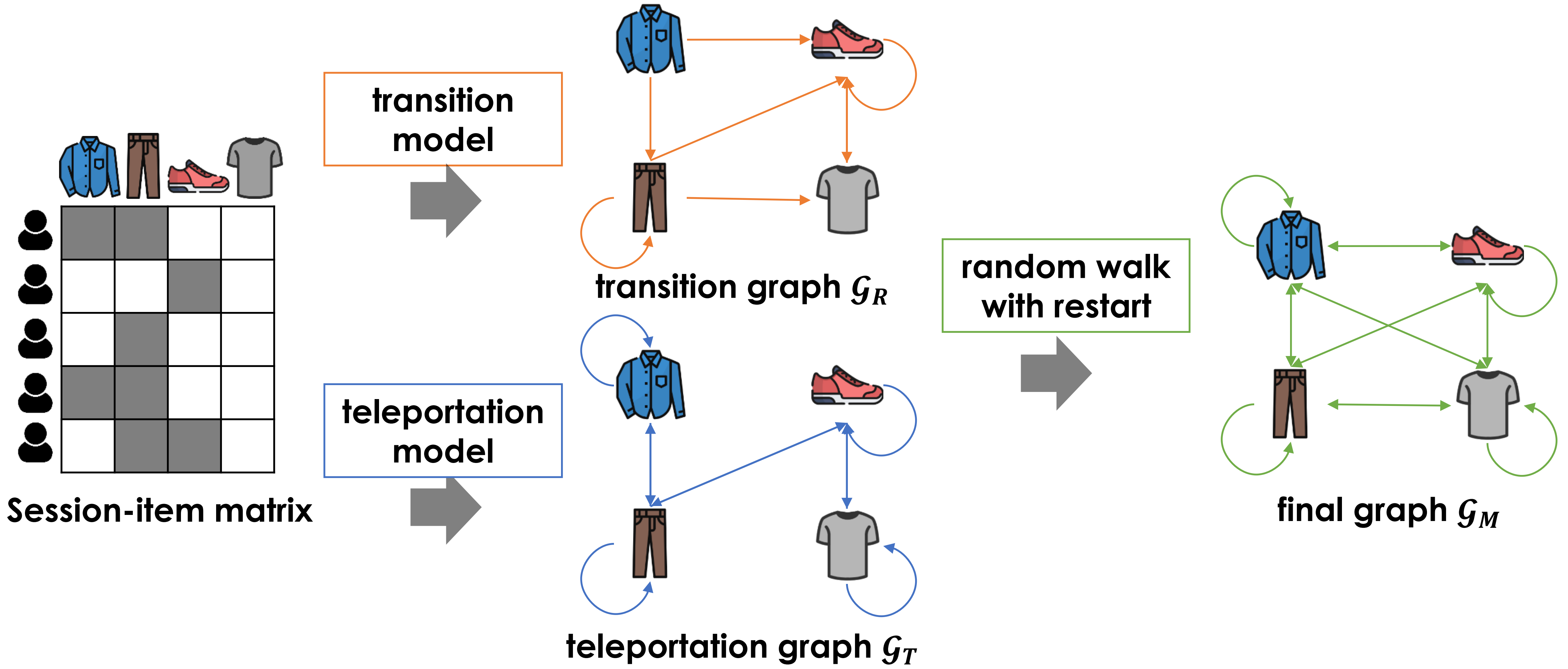
- **To build graphs for random walks, we devise linear item models.**
  - Each model can capture different characteristics of sessions.



# Proposed Model

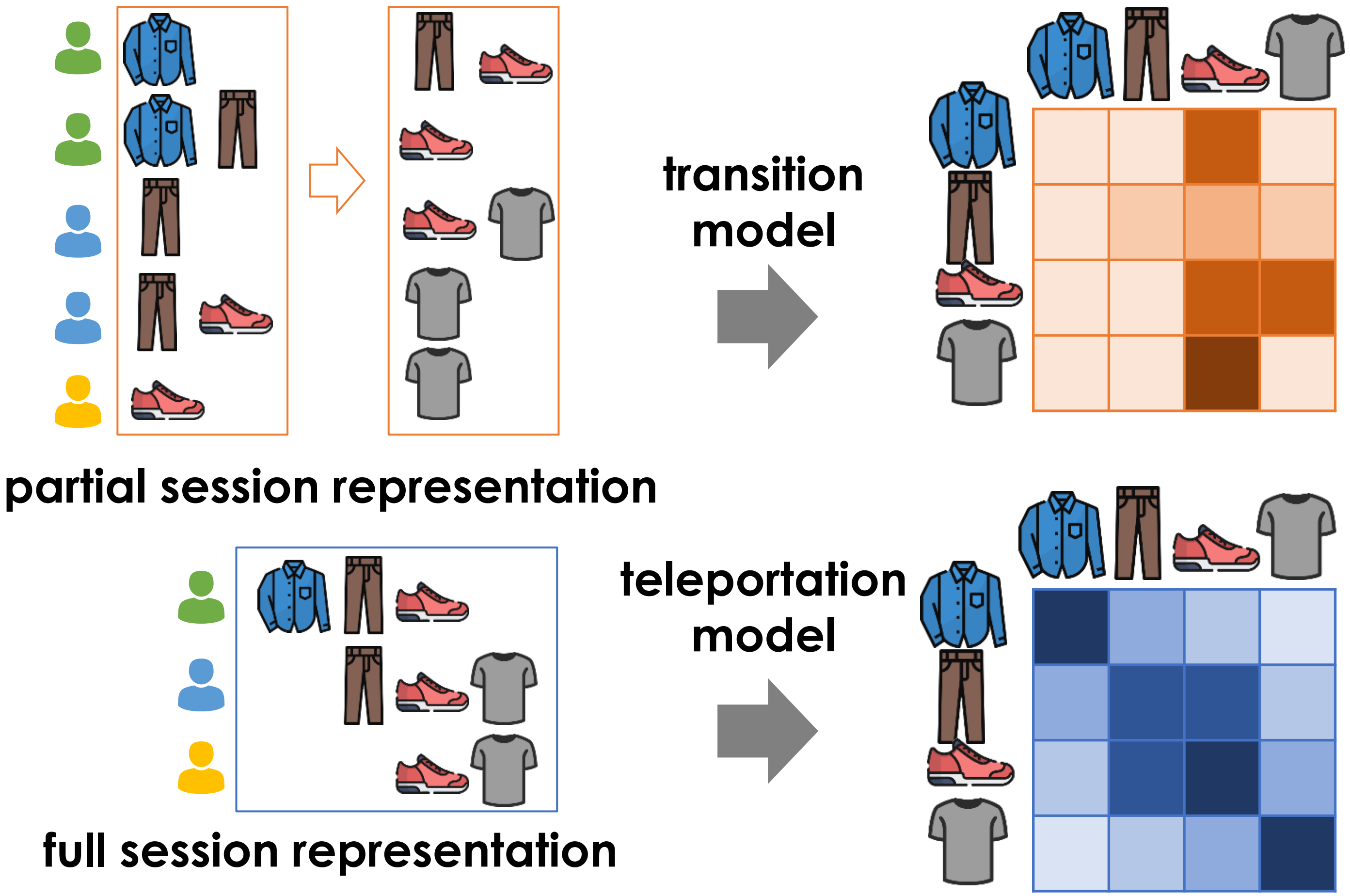
# Overview of the Proposed Model

- **We design two linear models and constitute a final item graph.**
  - The final graph is used for personalized recommendation.



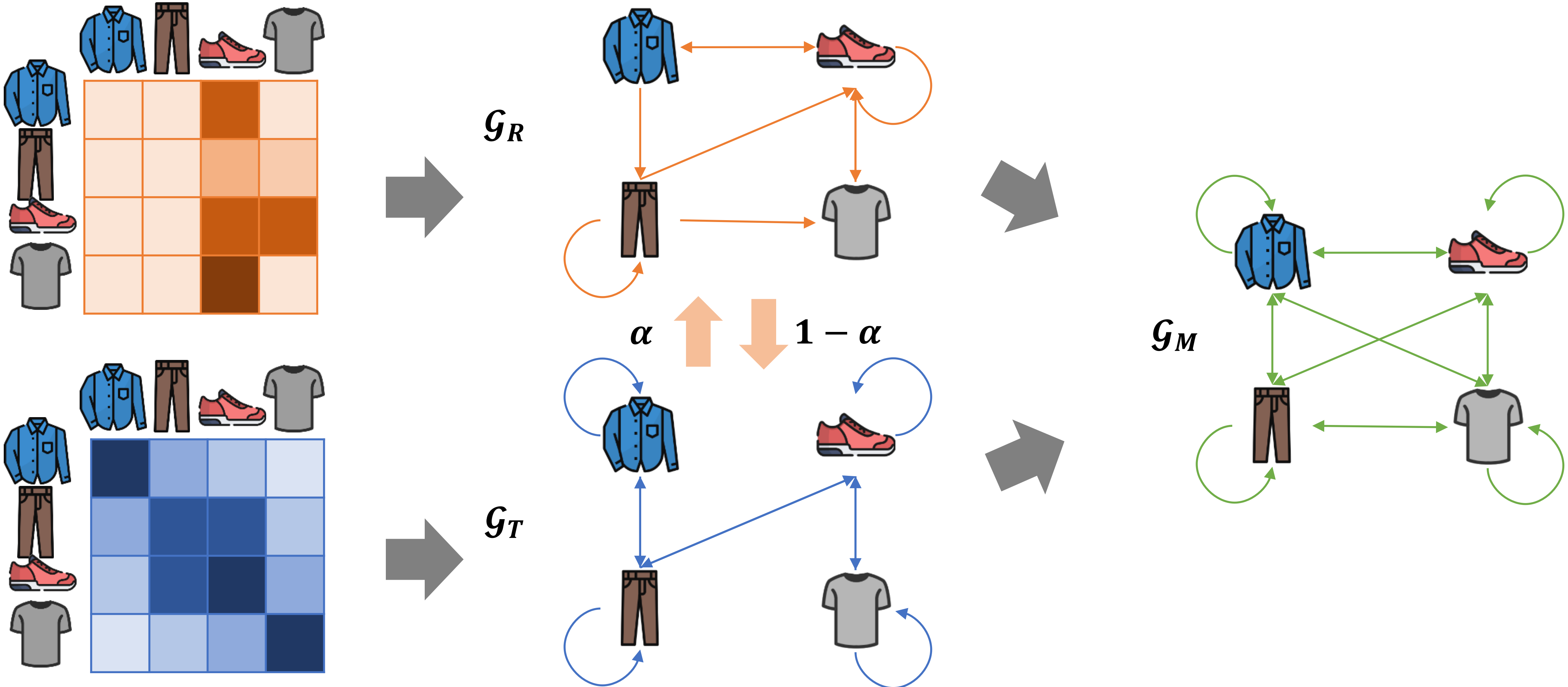
# Linear Item Models

➢ **We learn the linear models using two session representations.**
- Each captures the sequential dependency and item similarities.



# Random Walk with Restarts

- **We adopt the 'random walk with restarts' using the two graphs.**
  - A random walker jumps ($\boldsymbol{\alpha}$) or restarts ($\mathbf{1}$-$\boldsymbol{\alpha}$) on a node using two graphs; she could land on various nodes with certain probabilities.



# Detail: Training and Inference

## ➢ Model Training

- We utilize the power method to compute the stationary distribution.

$$x_{\infty} = \alpha^{\infty} x_{(0)} R^{\infty} + \sum_{k=0}^{\infty} \alpha^{k} (1-\alpha) x_{(0)} T R^{k}$$

$$\approx x_{(0)} \sum_{k=0}^{\infty} \alpha^{k} (1-\alpha) T R^{k} = x_{(0)} M$$

Teleportation matrix

Transition matrix

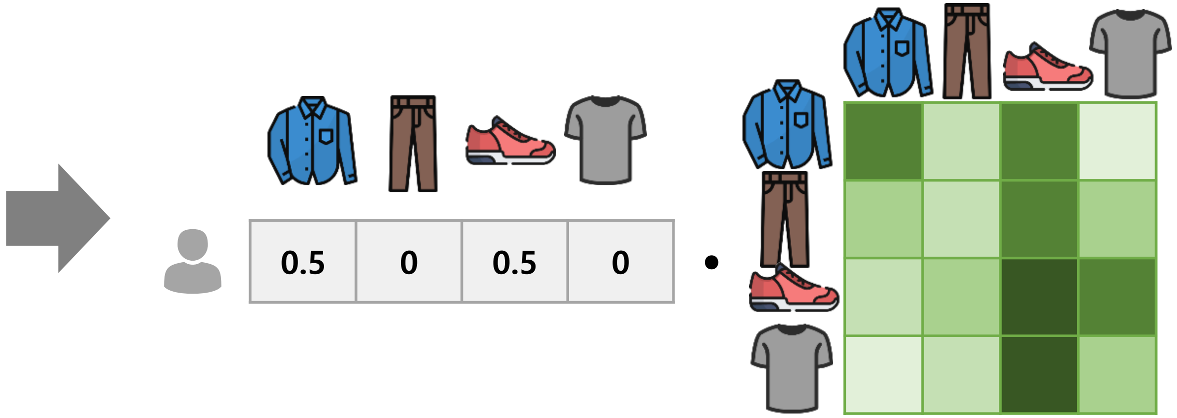
session vector

Trained final matrix

## ➢ Model inference

- For a new session, we compute the score using $\boldsymbol{x_{new}}$ and $\boldsymbol{M}$.

$$\hat{x} = x_{new}^{\top} M$$



# Experiments

# Experimental Setup: Dataset

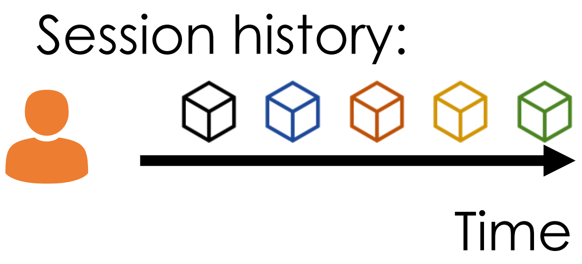
➢ **We evaluate the proposed model over public datasets.**
- For a fair comparison, we evaluate both on 1-split and 5-split datasets.

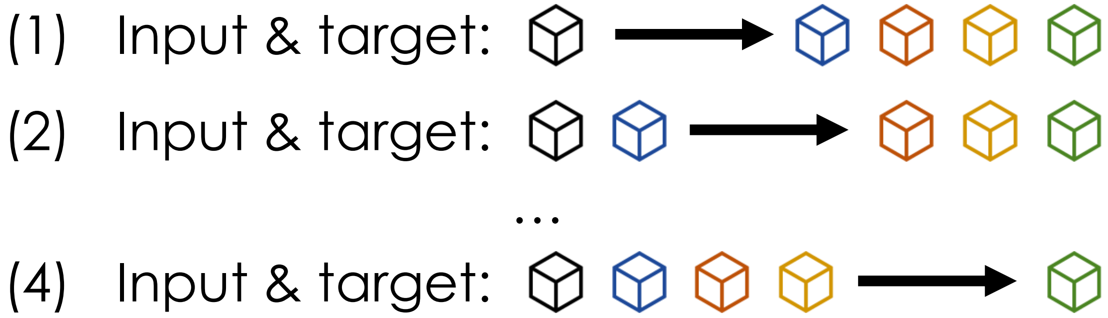
| Split | Dataset | # of actions | # of sessions | # of items |
|---|---|---|---|---|
| 1-split | YooChoose 1/4 (YC-1/4) | 7,909,307 | 1,939,891 | 30,638 |
| | DIGINETICA (DIGI1) | 916,370 | 188,807 | 43,105 |
| 5-split | YooChoose (YC5) | 5,426,961 | 1,375,128 | 28,582 |
| | DIGINETICA (DIGI5) | 203,488 | 41,755 | 32,137 |
| | RetailRocket (RR) | 212,182 | 59,962 | 31,968 |

# Evaluation Protocol and Metrics

## ➢Evaluation protocol: iterative revealing scheme

- We iteratively expose the item of a session to the model.




## ➢Evaluation metrics

- HR@20 and MRR@20
  - To predict only the next item in a session
- R@20 and MAP@20
  - To consider all subsequent items for a session

# Competitive Models

- **Two Non-Neural models**
  - STAN: an improved version of SKNN by considering sequence and time.
  - SLIST: a linear model designed for a session-based recommendation.

- **Five Neural models**
  - NARM: an improved version of GRU4REC+ using an attention mechanism.
  - STAMP: an attention-based model for capturing user's interests.
  - SR-GNN: a GNN-based model to capture complex dependency.
  - NISER+: an improved version of SR-GNN using normalized embeddings.
  - GCE-GNN: a GNN-based model considering inter-session relationships.

*Diksha Garg et. al., "Sequence and Time Aware Neighborhood for Session-based Recommendations: STAN", SIGIR 2019*
*Minjin Choi et. al., "Session-aware Linear Item-Item Models for Session-based Recommendation", WWW 2021*
*Jing Li et. al., "Neural Attentive Session-based Recommendation" CIKM 2017.*
*Qiao Liu et. al., "STAMP: ShortTerm Attention/Memory Priority Model for Session-based Recommendation" KDD 2018.*
*Shu Wu et. al., "Session-Based Recommendation with Graph Neural Networks", AAAI 2019.*
*Priyanka Gupta et. al., "NISER: Normalized Item and Session Representations with Graph Neural Networks", ArXiv 2019.*
*Ziyang Wang et. al, "Global Context Enhanced Graph Neural Networks for Session-based Recommendation", SIGIR 2020.*

# Accuracy: Ours vs. Competing Models

- **S-Walk shows competitive or state-of-the-art performances.**
  - It is challenging to achieve outstanding accuracy on all the datasets.

| Dataset | Metric | STAN (Non-Neural Models) | SLIST (Non-Neural Models) | NARM (Neural Models) | STAMP (Neural Models) | SR-GNN (Neural Models) | NISER+ (Neural Models) | GCE-GNN (Neural Models) | S-Walk$_{(1)}$ (Ours) | S-Walk (Ours) | Gain(%) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DIGI5 | R@20 | 0.3720 | 0.3803 | 0.3254 | 0.3040 | 0.3232 | 0.3727 | 0.3927 | 0.3761 | **0.3995** | 1.73 |
| | HR@20 | 0.4800 | 0.4915 | 0.4188 | 0.3917 | 0.4158 | 0.4785 | 0.5086 | 0.4873 | **0.5115** | 0.57 |
| RR | R@20 | 0.4748 | 0.4724 | 0.4526 | 0.3917 | 0.4438 | 0.4630 | 0.4841 | 0.4810 | **0.4994** | 3.16 |
| | HR@20 | 0.5938 | 0.5877 | 0.5549 | 0.4620 | 0.5433 | 0.5651 | 0.6007 | 0.6019 | **0.6226** | 3.65 |
| YC5 | R@20 | 0.4986 | 0.5122 | 0.5109 | 0.4979 | 0.5060 | 0.5146 | 0.4972 | 0.5096 | **0.5189** | 0.85 |
| | HR@20 | 0.6656 | 0.6867 | 0.6751 | 0.6654 | 0.6713 | 0.6858 | 0.6650 | 0.6834 | **0.6906** | 0.57 |
| NOWP | R@20 | 0.1696 | 0.1840 | 0.1274 | 0.1253 | 0.1400 | 0.1493 | 0.1504 | 0.1837 | **0.1915** | 4.08 |
| | HR@20 | 0.2414 | 0.2689 | 0.1849 | 0.1915 | 0.2113 | 0.2196 | 0.2122 | 0.2678 | **0.2693** | 0.15 |
| YC-1/4 | R@20 | 0.4952 | 0.5130 | 0.5097 | 0.5008 | 0.5095 | 0.5164 | 0.5030 | 0.5103 | **0.5213** | 0.95 |
| | HR@20 | 0.6846 | 0.7175 | 0.7079 | 0.7021 | 0.7118 | 0.7182 | 0.7036 | 0.7145 | **0.7204** | 0.31 |

S-Walk$_{(1)}$ is a variant of S-Walk trained only up to the first step.

# Scalability: Ours vs. Competing Models

- **S-Walk shows faster inference time thanks to its simpler structure.**
  - This property is highly desirable for deploying S-Walk to real-world applications.

| Models | YC-1/4 | | DIGI5 | | RR | |
|---|---|---|---|---|---|---|
| | GFLOPs | Time(s) | GFLOPs | Time(s) | GFLOPs | Time(s) |
| SR-GNN (in GPU) | 1282.8 | 70.8 | 765.4 | 49.2 | 247.2 | 12.7 |
| NISER+ (in GPU) | 2605.8 | 87.1 | 1551.0 | 59.7 | 501.8 | 15.7 |
| GCE-GNN (in GPU) | 51094.8 | 108.8 | 10445.9 | 47.0 | 9446.0 | 19.8 |
| S-Walk (in CPU) | **11.0** | **20.5** | **4.9** | **8.3** | **2.3** | **5.2** |
| Gain (S-Walk vs. GCE-GNN) | 4632.3x | 5.3x | 2131.3x | 8.9x | 4133.2x | 3.8x |

# Ablation Study: Component of Random Walks

- The **complete S-Walk shows the best performance** compared to using other models as the transition or teleportation models.

| Transition model | Teleportation model | YC-1/4 | | DIGI5 | | RR | |
|---|---|---|---|---|---|---|---|
| | | R@20 | MAP@20 | R@20 | MAP@20 | R@20 | MAP@20 |
| SR | I | 0.5109 | 0.0394 | 0.3809 | 0.0260 | 0.4812 | 0.0291 |
| | AR | 0.4952 | 0.0378 | 0.3879 | 0.0266 | 0.4817 | 0.0291 |
| | **Ours** | **0.5171** | **0.0400** | **0.3930** | **0.0270** | **0.4950** | **0.0301** |
| **Ours** | I | 0.5175 | 0.0399 | 0.3808 | 0.0259 | 0.4826 | 0.0292 |
| | AR | 0.5009 | 0.0383 | 0.3899 | 0.0268 | 0.4856 | 0.0293 |
| | **Ours** | **0.5205** | **0.0403** | **0.3936** | **0.0271** | **0.4979** | **0.0303** |

# Conclusion

# Conclusion

- **We propose S-Walk, a session-based recommendation using random walks.**
  - It can fully capture intra-session and inter-session correlations in sessions.

- **S-Walk achieves competitive or state-of-the-art accuracy.**
  - It is challenging to achieve outstanding performance over various datasets consistently.

- **S-Walk shows high scalability and fast inference speed.**
  - The inference of S-Walk using CPU is up to 8.9x faster than DNN models using GPU.
  - S-Walk can be compressed highly robustly, without sacrificing its accuracy.

# Q&A


Email: zxcvxd@skku.edu

Code: https://github.com/jin530/SWalk