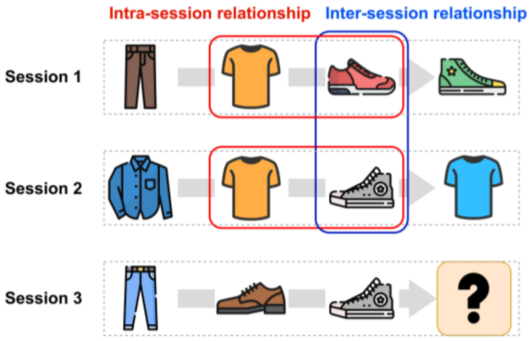
S-Walk: Accurate and Scalable Session-based Recommendation with Random Walks

Minjin Choi[1], Jinhong Kim[2], Joonseok Lee[3,4], Hyunjung Shim[5], Jongwuk Lee[1]

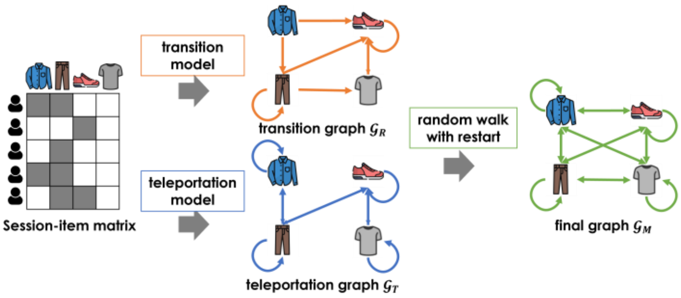
Sungkyunkwan University[1], Naver Corp.[2], Google Research[3], Seoul National University[4], Yonsei University[5]

Motivation



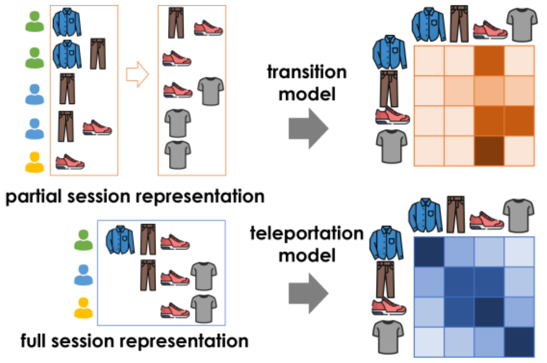
- **Session-based Recommendation (SR)** predicts the next item(s) based on the current session
- Existing SR models pay less attention to **inter-session relationships of items**, which has the potential to improve accuracy.
- DNN-based SR models suffer **computational efficiency and scalability** issues.
- We utilize **random walk methods** to exploit intra- and inter-session relationships.
- To build graphs for random walks, we devise **linear item models**; each model can capture different characteristics of sessions.

Proposed method: S-Walk



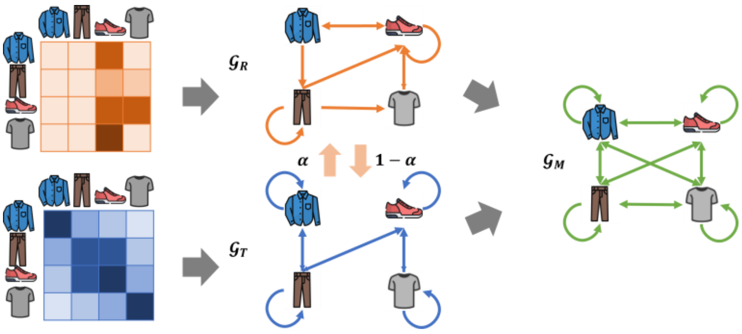
- We propose a novel session-based recommendation with a random walk, **S-Walk**.
- S-Walk effectively captures **intra- and inter-session correlations** by handling high-order relationships among items using **random walks with restart (RWR)**.
- By adopting linear models with closed-form solutions for random walks graph, S-Walk is **highly efficient and scalable**.

Linear Item Models



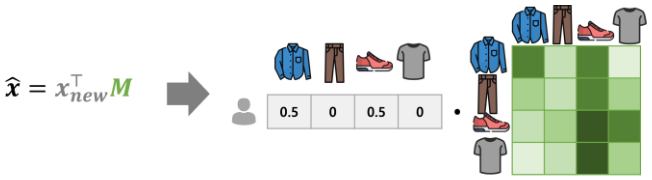
- We learn the **linear models** using two session representations; each captures the sequential dependency and item similarities.
- **Each model produces its relevance matrix** over the **transition graph** and the **teleportation graph**, where each node corresponds to an item and an edge indicates the relevance between a pair of items.

Random Walk with Restarts



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

Model Inference



- For a new session, we compute the score using x_new and M.
- It is merely a sparse matrix multiplication.

Experimental setup and results

- We evaluate the proposed model over **public datasets**; for a fair comparison, we evaluate both on 1-split and 5-split datasets.
- We use an **iterative revealing scheme**, that is, iteratively expose the item of a session to the model
- For evaluation metrics, we use **HR@20 and MRR@20** to predict only the next item in a session and **R@20 and MAP@20** to consider all subsequent items for a session

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

- S-Walk shows **competitive or state-of-the-art performances**, even though It is challenging to be outstanding on all the datasets.

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

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

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