

Point-of-Interest Recommendation Systems in Location-Based Social Networks

Tony Song



LBSN? POI?

Location-based social networks (**LBSNs**):

Social networks tied with geographical information

Point-of-interest (**POI**):

A specific location where users show interest (restaurant, bar, cinema, etc.)

The screenshot shows a Facebook search for "kyoto sushi". The top navigation bar includes "All", "Posts", "People", "Photos", "Videos", "Pages", "Places", "Groups", "Apps", "Events", and "Links". The "Places" tab is selected. On the left, there are filters for "SOCIAL" (Visited by friends, Good for groups), "HOURS" (Open Now), and "PRICE" (\$, \$\$, \$\$\$, \$\$\$\$).

Two restaurant listings are shown:

- 1 Kyoto Sushi Restaurant**: 4.4 stars (27 reviews), Sushi Restaurant, \$\$\$\$ price range. Address: 2841 Lyndale Ave S, Minneapolis, Minnesota. Phone: (612) 870-9999. Status: Opens tomorrow. Description: Jubair, Momoko and 4 other friends have been here. People talk about all you can eat sushi. A review by Morgan Johnson (5.0 stars, over a year ago) says: "Service is amazing but OMG the food is to die for! Warning, it's expensive but you get great food for the price!!".
- 2 Kyoto Sushi**: 4.4 stars (56 reviews), Sushi Restaurant, \$\$\$\$ price range.

A map of Minneapolis on the right shows the locations of these restaurants marked with red pins. The map includes labels for various neighborhoods like Willard-Hay, Near North, Nicollet Island, Harb Holm, Bryn Mawr, Kenwood, East Isles, Phillips West, West Calhoun, Ecco, Lyndale, Powderhorn Park, Linden Hills, Lyndale Park, Bryant, Regina, Northrup, Tangletown, Page, Hale, York Park, Armatage, Keny, and Windom.



Motivation - Big question

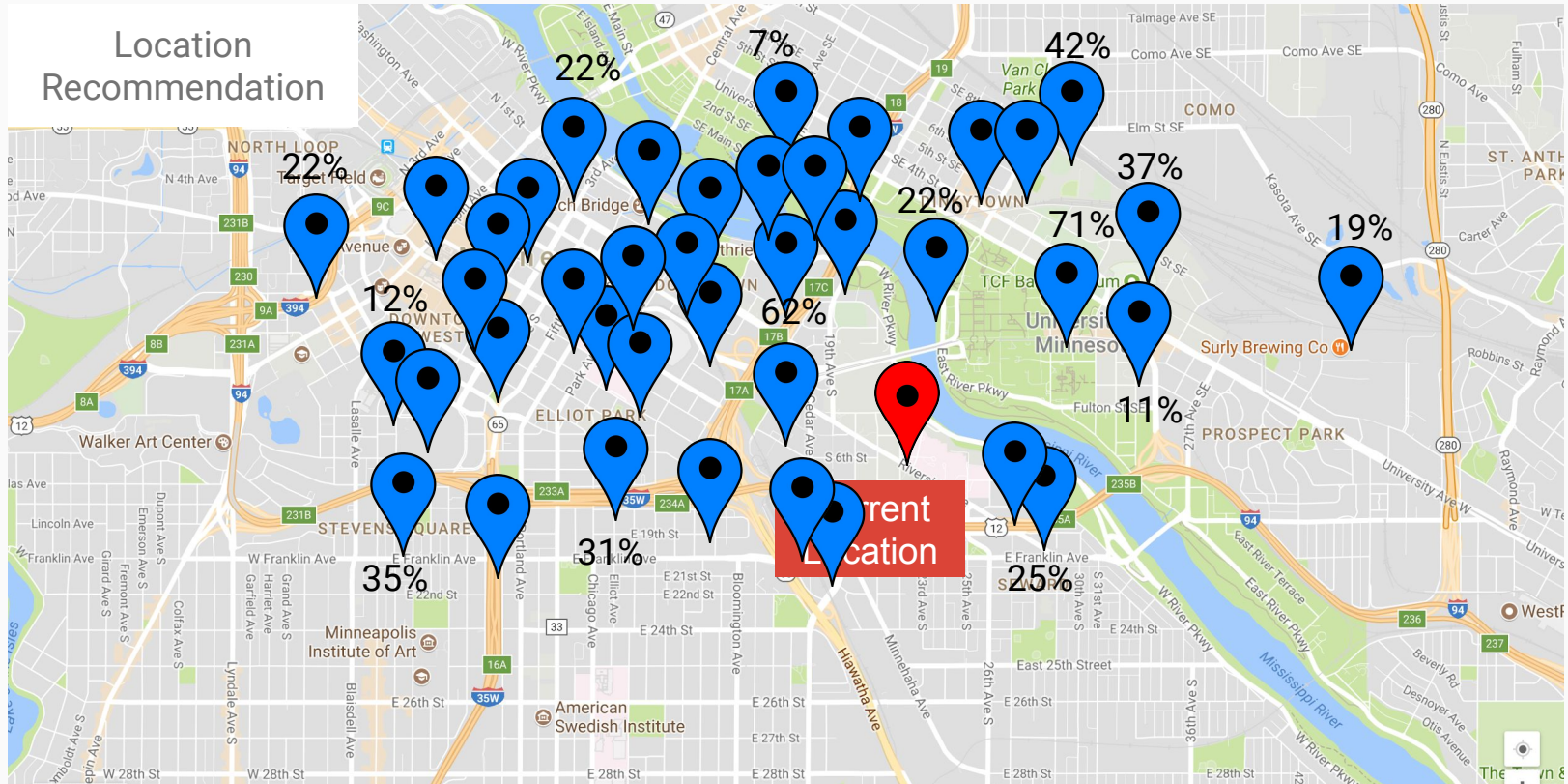


Table of Contents

- Backgrounds
 - Tensor
 - Matrix Factorization
 - Markov Chain
- Algorithms
 - FPMC
 - FPMC-LR
 - TAD-FPMC
- Performance Analysis

Table of Contents

- **Backgrounds**
 - **Tensor**
 - **Matrix Factorization**
 - **Markov Chain**
- Algorithms
 - FPMC
 - FPMC-LR
 - TAD-FPMC
- Performance Analysis

Tensor

- Generalization of vectors
- Multi-dimensional arrays in Java

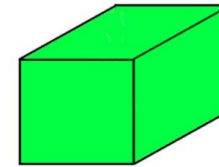
1D TENSOR /
VECTOR

5
7
45
12
-6
3
22
1
6
3
-9

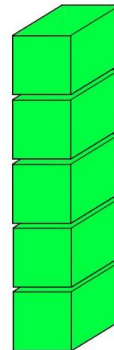
2D TENSOR /
MATRIX

-9	4	2	5	7
3	0	12	8	61
1	23	-6	45	2
22	3	-1	72	6

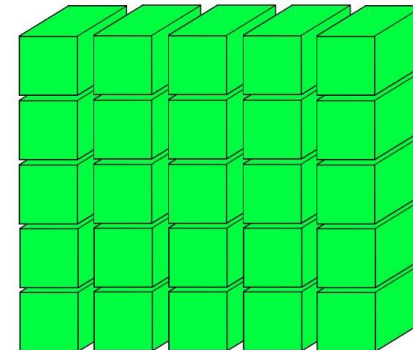
3D TENSOR /
CUBE



-9	4	2	5	7
3	0	12	8	61
1	23	-6	45	2
22	3	-1	72	6



4D TENSOR
VECTOR OF CUBES

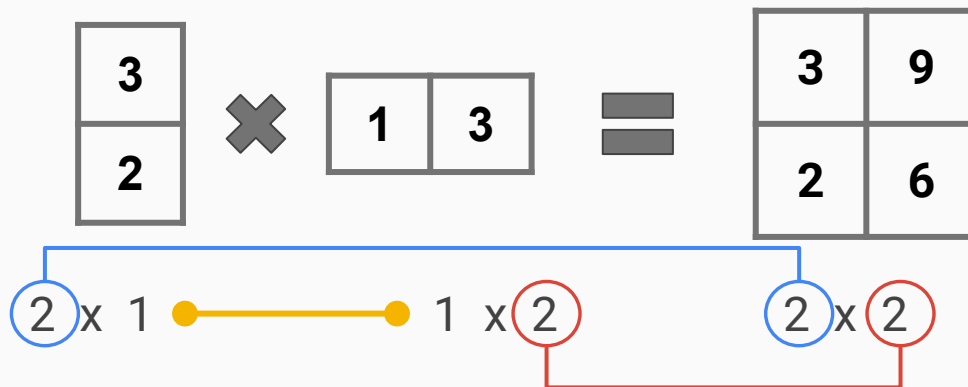


5D TENSOR
MATRIX OF CUBES

Matrix Factorization (MF)

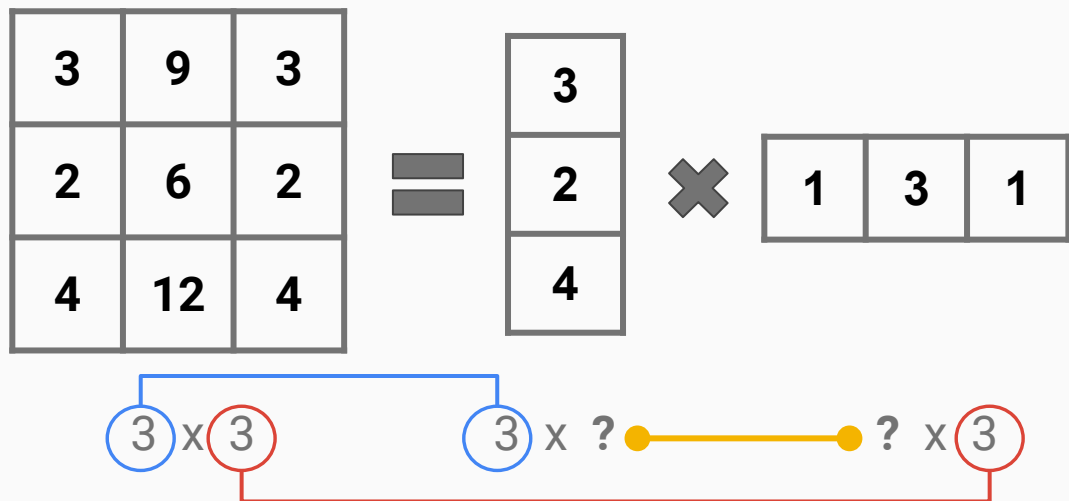
Review:

Matrix multiplication



Matrix Factorization (MF)

A technique to factor a matrix as a product of multiple matrices



Matrix Factorization (MF)

Question:

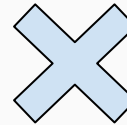
Given a matrix of user ratings for different locations, **predict empty ratings** $0 (n/a)$

Location\User	User 1	User 2	User 3
Tony's Sushi	5	4	$0 (n/a)$
Theatre	3	3	1
Dollar Store	$0 (n/a)$	$0 (n/a)$	2

Matrix Factorization (MF)

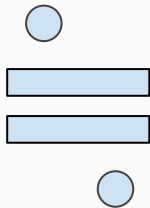
Matrix L

Location\Feature	<i>shopping</i>	<i>food</i>
Tony's Sushi	?	?
Theatre	?	?
Dollar Store	?	?



Matrix U

Feature\User	User 1	User 2	User 3
<i>shopping</i>	?	?	?
<i>food</i>	?	?	?

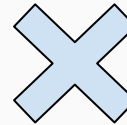


Location\User	User 1	User 2	User 3
Tony's Sushi	5	4	0 (<i>n/a</i>)
Theatre	3	3	1
Dollar Store	0 (<i>n/a</i>)	0 (<i>n/a</i>)	2

Matrix Factorization (MF)

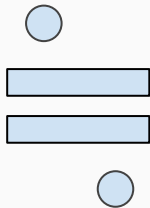
Matrix L

Location\Feature	<i>shopping</i>	<i>food</i>
Tony's Sushi	1	5
Theatre	2	1
Dollar Store	5	2



Matrix U

Feature\User	User 1	User 2	User 3
<i>shopping</i>	1	1	0.1
<i>food</i>	0.8	0.6	0.75

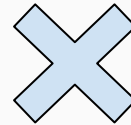


Location\User	User 1	User 2	User 3
Tony's Sushi	5	4	0 (<i>n/a</i>)
Theatre	3	3	1
Dollar Store	0 (<i>n/a</i>)	0 (<i>n/a</i>)	2

Matrix Factorization (MF)

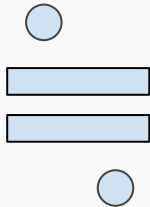
Matrix L

Location\Feature	<i>shopping</i>	<i>food</i>
Tony's Sushi	1	5
Theatre	2	1
Dollar Store	5	2



Matrix U

Feature\User	User 1	User 2	User 3
<i>shopping</i>	1	1	0.1
<i>food</i>	0.8	0.6	0.75



Location\User	User 1	User 2	User 3
Tony's Sushi	5	4	$1 * 0.1 + 5 * 0.75 = 4$
Theatre	3	3	1
Dollar Store	0 (n/a)	0 (n/a)	2

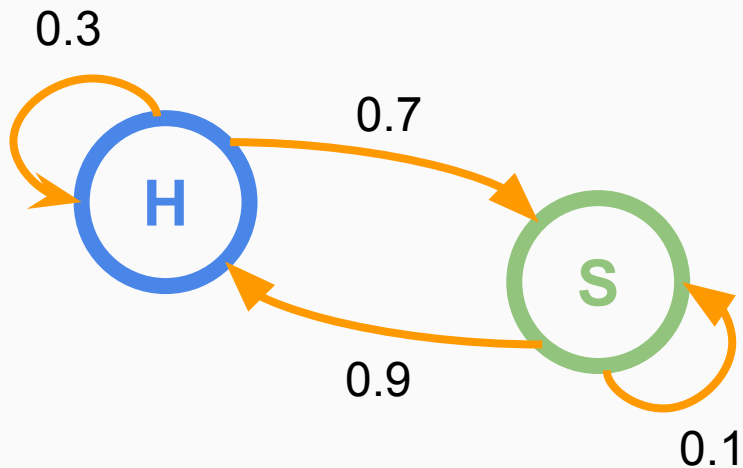
Matrix Factorization (MF)

What's the point?

Based on the calculated predictions, we can make **recommendations**.

Location\User	User 1	User 2	User 3
Tony's Sushi	5	4	4
Theatre	3	3	1
Dollar Store	5	5	2

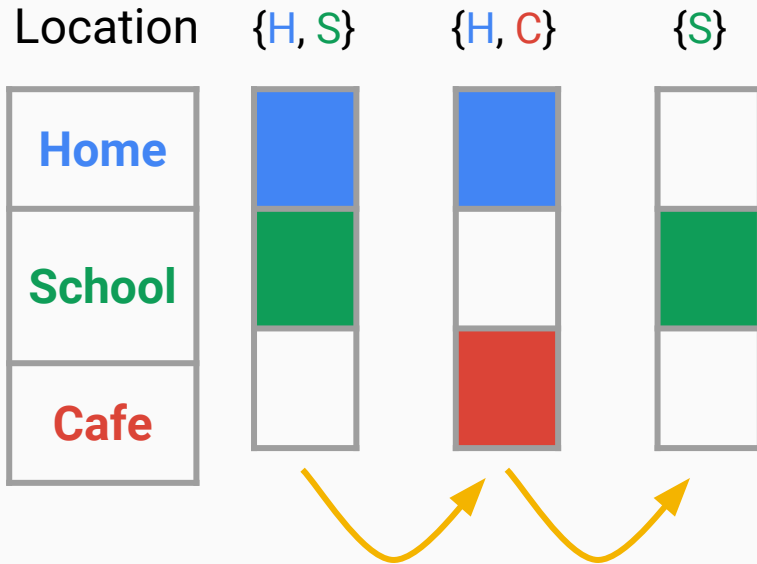
Markov Chain (MC)



Stochastic model to represent different states (locations) and their transitional possibilities.

	Home	School
Home	0.3	0.7
School	0.9	0.1

Markov Chain (MC) for Sets



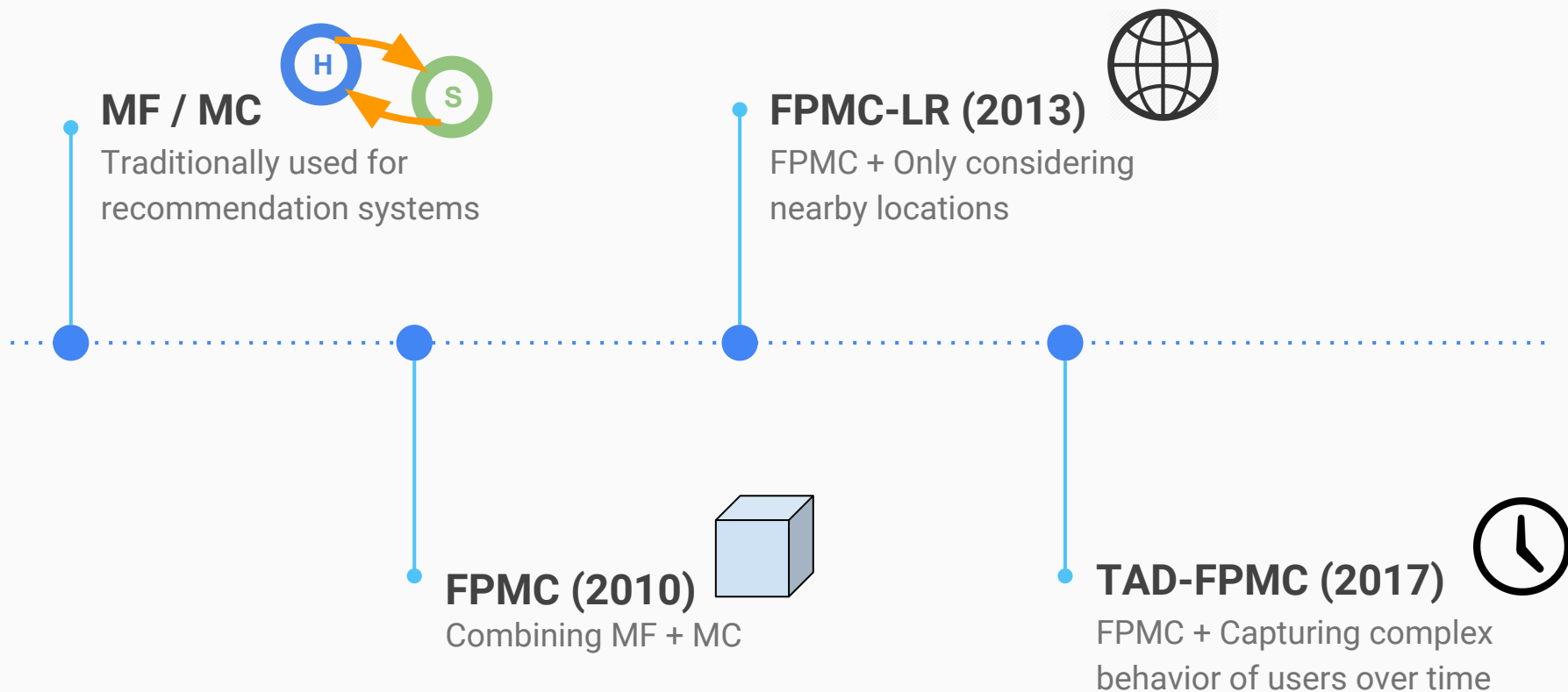
Each state is now a set of locations instead of a single location.

	Home	School	Cafe	#
Home	1/2	1/2	1/2	2
School	1/1	0/1	1/1	1
Cafe	0/1	1/1	0/1	1

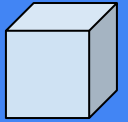
Table of Contents

- Backgrounds
 - Tensor
 - Matrix Factorization
 - Markov Chain
- **Algorithms**
 - **FPMC**
 - **FPMC-LR**
 - **TAD-FPMC**
- Performance Analysis

POI Recommendation History over Time



Factorized Personalized Markov Chain (FPMC)



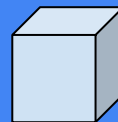
Problem of Markov chain

One Markov chain is used for all users (Recommendation is not personalized)



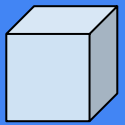
How FPMC solves the problem

Having a Markov chain for each user

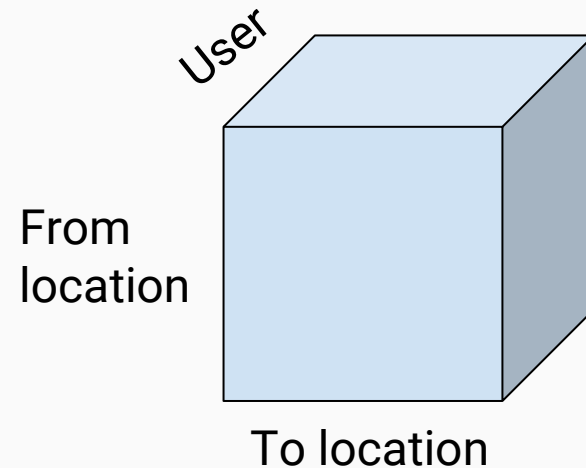
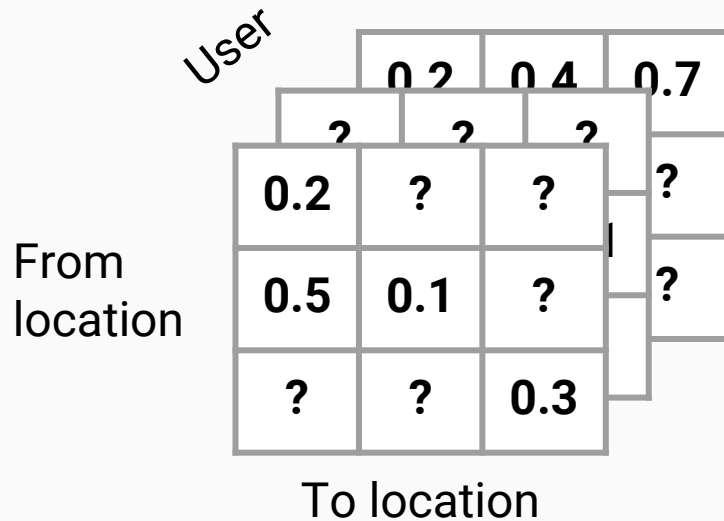


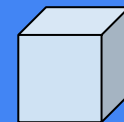
- 1. Make a Markov chain for each user**
2. Stack them one onto another
3. Factor the cubic tensor using tensor factorization
4. Calculate predictions using the factors

	User 1	User 2	User 3																											
From location	<table border="1"><tr><td>0.2</td><td>?</td><td>?</td></tr><tr><td>0.5</td><td>0.1</td><td>?</td></tr><tr><td>?</td><td>?</td><td>0.3</td></tr></table>	0.2	?	?	0.5	0.1	?	?	?	0.3	<table border="1"><tr><td>?</td><td>?</td><td>?</td></tr><tr><td>0.3</td><td>?</td><td>0.1</td></tr><tr><td>0.6</td><td>?</td><td>?</td></tr></table>	?	?	?	0.3	?	0.1	0.6	?	?	<table border="1"><tr><td>0.2</td><td>0.4</td><td>0.7</td></tr><tr><td>?</td><td>0.1</td><td>?</td></tr><tr><td>?</td><td>0.5</td><td>?</td></tr></table>	0.2	0.4	0.7	?	0.1	?	?	0.5	?
0.2	?	?																												
0.5	0.1	?																												
?	?	0.3																												
?	?	?																												
0.3	?	0.1																												
0.6	?	?																												
0.2	0.4	0.7																												
?	0.1	?																												
?	0.5	?																												
To location																														

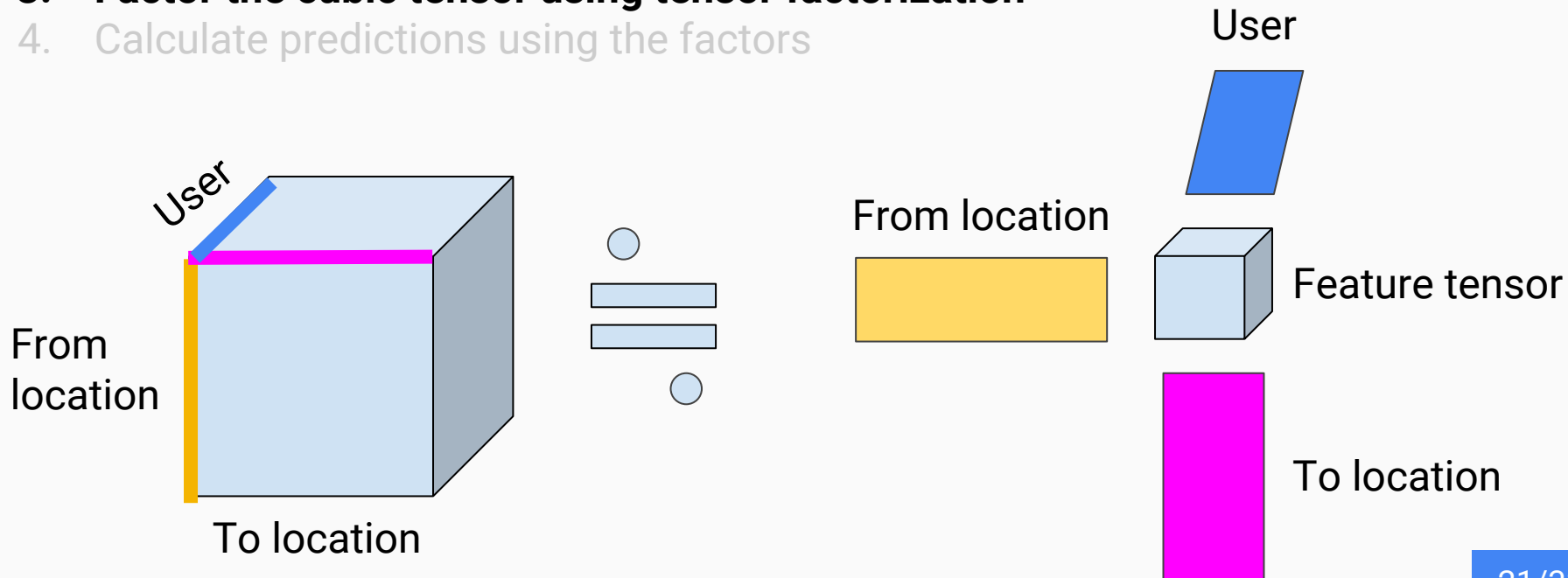


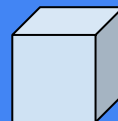
1. Make a Markov chain for each user
- 2. Stack them one onto another**
3. Factor the cubic tensor using tensor factorization
4. Calculate predictions using the factors





1. Make a Markov chain for each user
2. Stack them one onto another
- 3. Factor the cubic tensor using tensor factorization**
4. Calculate predictions using the factors

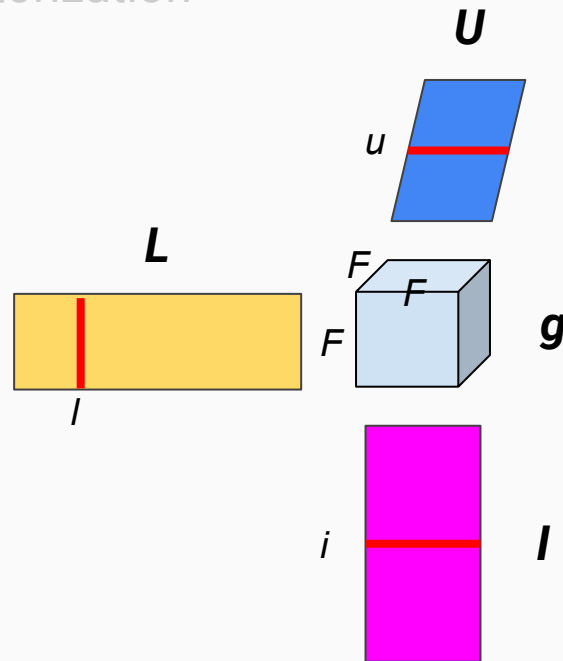




1. Make a Markov chain for each user
2. Stack them one onto another
3. Factor the cubic tensor using tensor factorization
- 4. Calculate predictions using the factors**

Probability of user \underline{u} going from location \underline{l} to location \underline{i}

$$x_{l,i,u} \approx \sum_{p=1}^F \sum_{q=1}^F \sum_{r=1}^F g_{pqr} L_{lp} I_{iq} U_{ur}$$



FPMC with Localized Region Constraint (FPMC-LR)



Problems of FPMC

Generic method for any
recommendation systems

Computationally expensive



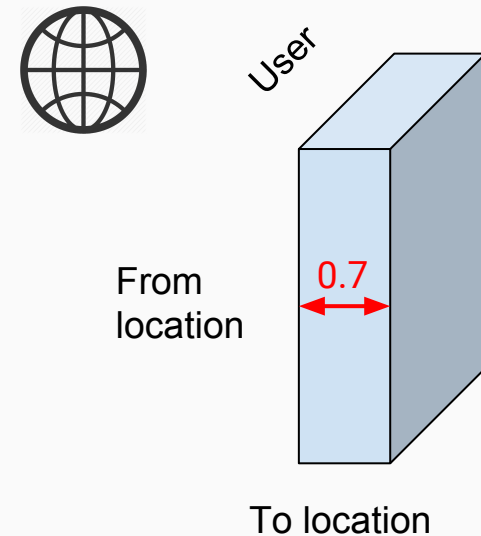
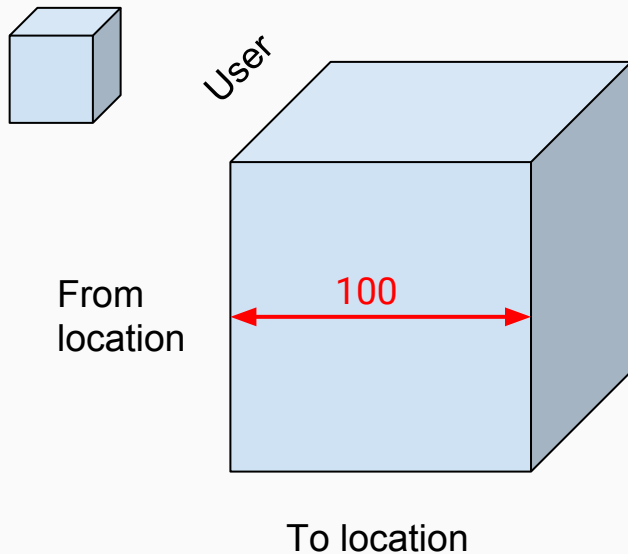
How FPMC-LR solves the problem

POI-specific method

Only consider nearby locations when
predictions are made



1. Make a Markov chain for each user
2. Stack them one onto another
3. Factor the **cubic tensor** using tensor factorization
4. Calculate predictions using the factors



Time-Aware Decaying FPMC (TAD-FPMC)

Problem of FPMC-LR

Complex user behavior over time is not incorporated



How TAD-FPMC solves the problem

Adding time variable when calculating the recommendations



1. Make a Markov chain for each user
2. Stack them one onto another
3. Factor the **cubic tensor** using tensor factorization
4. Calculate predictions using the factors

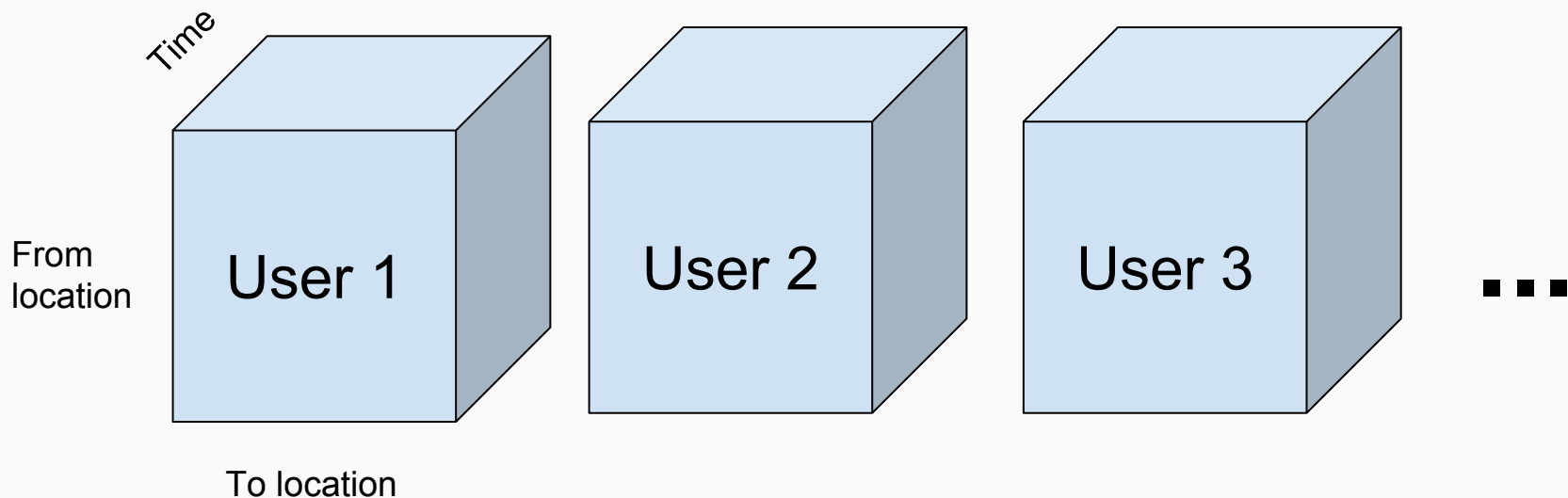
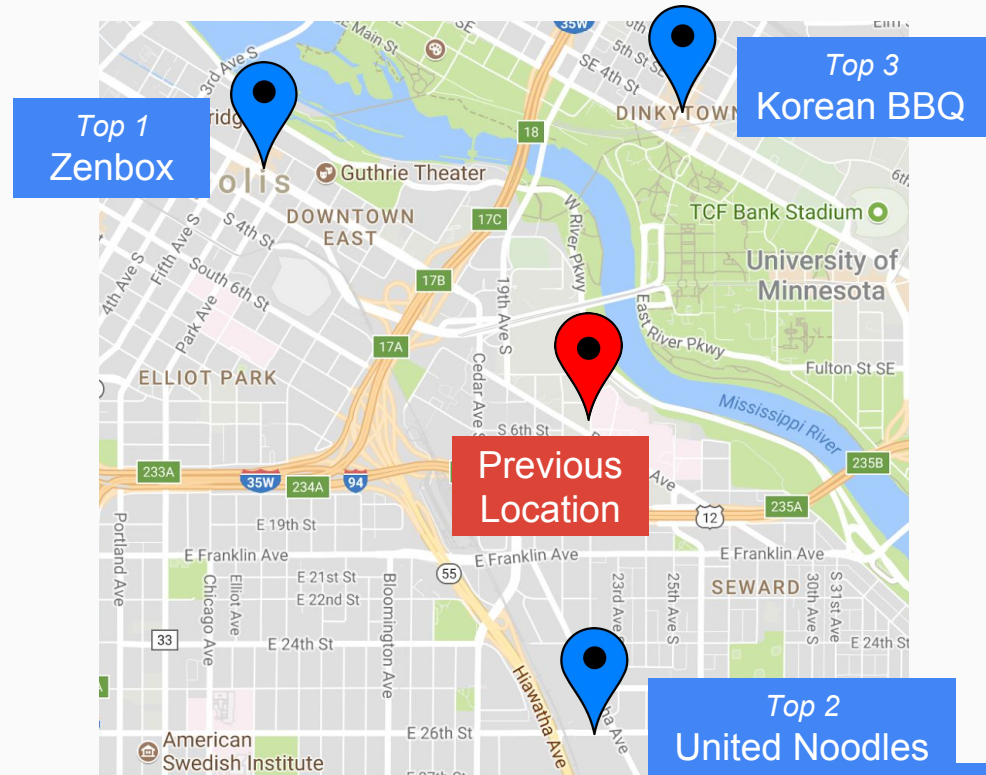


Table of Contents

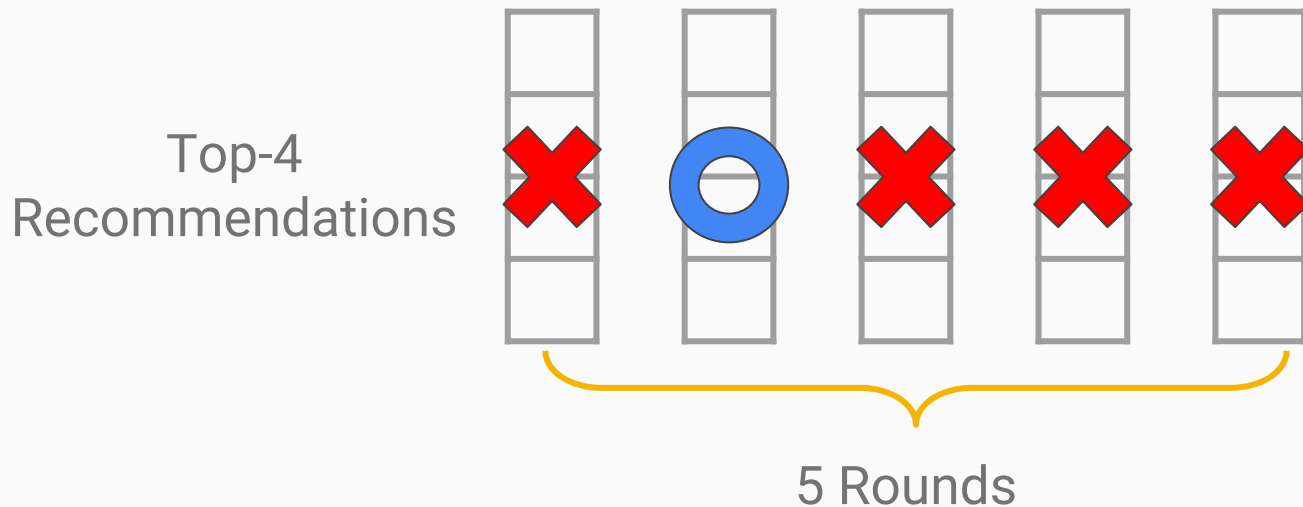
- Backgrounds
 - Tensor
 - Matrix Factorization
 - Markov Chain
- Algorithms
 - FPMC
 - FPMC-LR
 - TAD-FPMC
- **Performance Analysis**

1. Each Algorithm recommends a list of top- N places.
2. Recommendation is correct if the user indeed visited any place in the list at time t .

Top-3 recommendations

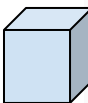
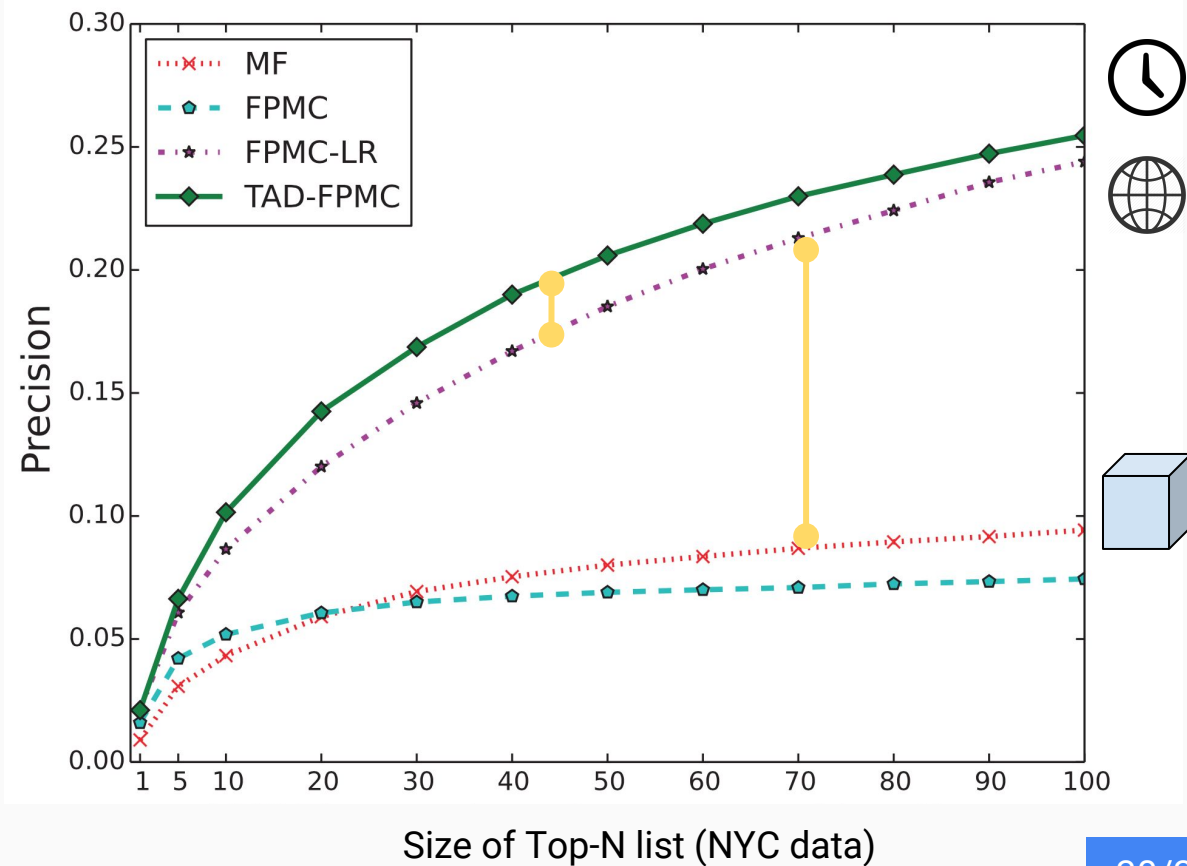


$$\text{Precision} = \frac{\# \text{ of correct predictions}}{\# \text{ of recommendation rounds}} = \frac{1}{5}$$

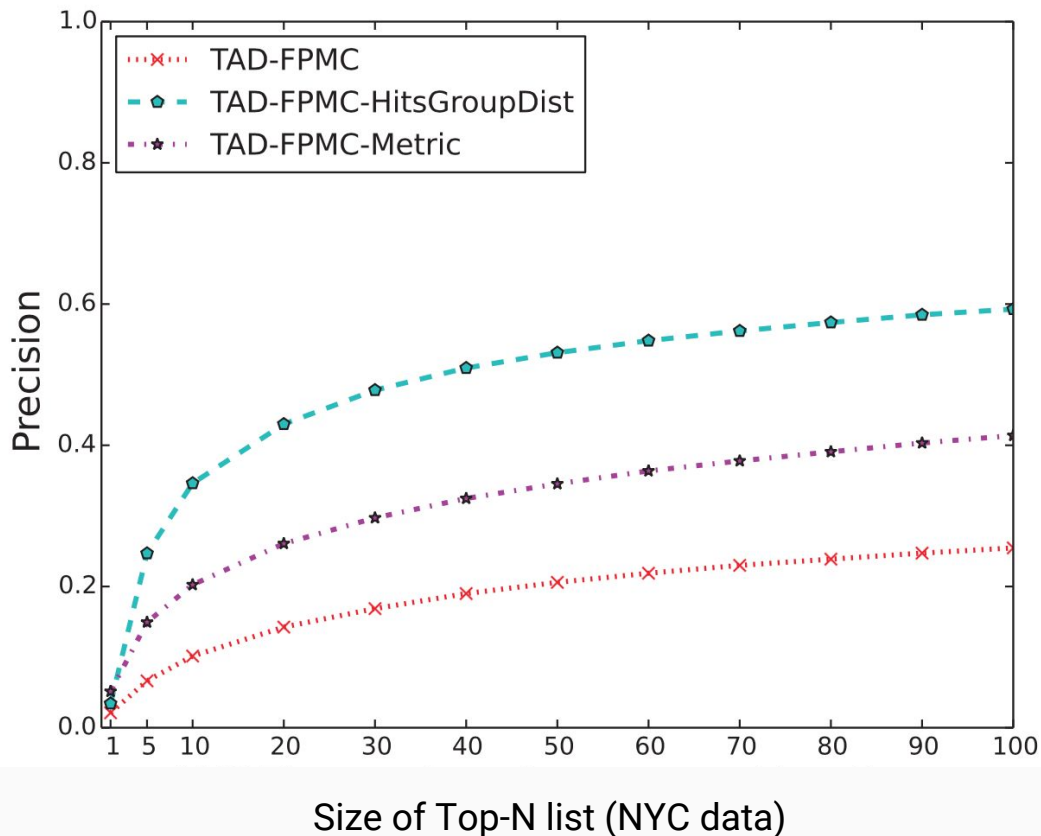


Performance Analysis

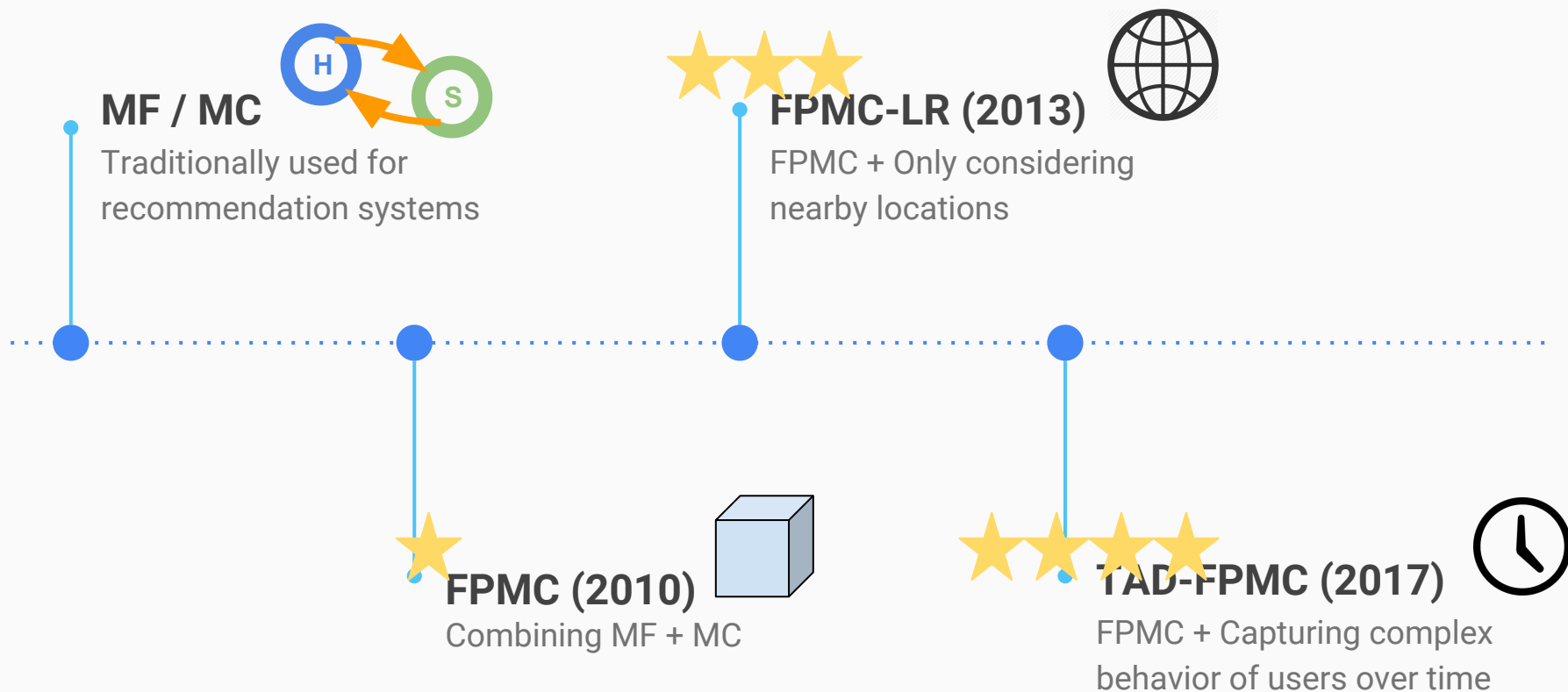
POI specific models
(FPMC-LR & TAD-FPMC)
show much better
performance



Variations of TAD-FPMC
far outperform all
existing models



Conclusion



Questions?

Contact

Tony Song

songx823@morris.umn.edu

<https://github.com/frogrammer>

References

- [1] C. Cheng, H. Yang, M. R. Lyu, and I. King. Where you like to go next: Successive point-of-interest recommendation. In Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, IJCAI '13, pages 2605–2611. AAAI Press, 2013.
- [2] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, Aug. 2009.
- [3] X. Li, M. Jiang, H. Hong, and L. Liao. A time-aware personalized point-of-interest recommendation via high-order tensor factorization. *ACM Trans. Inf. Syst.*, 35(4):31:1–31:23, June 2017.
- [4] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In Proceedings of the 19th International Conference on World Wide Web, WWW '10, pages 811–820, New York, NY, USA, 2010. ACM.
- [5] S. Rendle and L. Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In Proceedings of the Third ACM International Conference on Web Search and Data Mining, WSDM '10, pages 81–90, New York, NY, USA, 2010. ACM.