

# LARS: A Location-Aware Recommender System

Authors: Justin J. Levandoski, Mohamed Sarwat, Ahmed Eldawy,  
Mohamed F. Mokbel

Presenter: Hyunwoo Jung



SEOUL NATIONAL UNIVERSITY

# Motivation

- How to build an efficient location-aware recommendation system?

The screenshot displays the Foursquare City Guide interface for Seoul. The top navigation bar includes the Foursquare logo, a search bar with the text "I'm looking for...", and a "Current Map View" dropdown. On the right side of the bar are "Log In" and "Sign Up" buttons. Below the navigation bar, the page is titled "Suggestions for Best Nearby near Seoul". A filter bar contains buttons for "Specials", "Haven't Been", "Following", "Price", "Open Now", "Saved", and "Liked". A light blue banner encourages users to "Discover places that your friends and experts love" by creating an account. Below this banner are two buttons: "Sign up with Facebook" and "Sign up with email". The main content area on the left lists two recommendations:

- 1. Namsan Walking Trail** (9.6 rating)  
(남산산책로) Trail  
용신구 남산공원길 (남산 북측순환로), Seoul  
Save  
Young-Jin C. • March 1, 2015  
Great for walking and running.
- 2. Banpo Hangang Park** (9.5 rating)  
(반포한강공원) Park  
서초구 신반포로11길 40, Seoul  
Save

The right side of the screenshot shows a map of Seoul with 30 numbered blue location markers scattered across the city, primarily following the Han River and surrounding urban areas. The map includes standard navigation controls like zoom in (+), zoom out (-), and a search area button.



# Related Works

- Location-based services
  - Ex1) "local favorites" of Netflix
  - Ex2) Hyper-local place ranking  
(user location, location-related-query)  $\Rightarrow$  top points of interest

$\Rightarrow$  Doesn't provide personalized recommendations.
- Geo-measured friend-based collaborative filtering  
 $\Rightarrow$  Large-scale real-world deployment is not considered.



# Types of Location-based Ratings

- Spatial ratings for non-spatial items  
(user, **user location**, rating, item)  
ex) A user located at home rating a book.
- Non-spatial ratings for spatial items  
(user, rating, item, **item location**)  
ex) A user with unknown location rating a restaurant.
- Spatial ratings for spatial items  
(user, user location, rating, item, item location)  
ex) A user at his/her office rating a restaurant visited for lunch.



# Observation: Preference Locality

- Users in a region share interests.

U.S. State	Top Movie Genres	Avg. Rating
Minnesota	Film-Noir	3.8
	War	3.7
	Drama	3.6
	Documentary	3.6
Wisconsin	War	4.0
	Film-Noir	4.0
	Mystery	3.9
	Romance	3.8
Florida	Fantasy	4.3
	Animation	4.1
	War	4.0
	Musical	4.0

(a) Movielens preference locality

Users from:	Visited venues in:	% Visits
Edina, MN	Minneapolis , MN	37 %
	Edina , MN	59 %
	Eden Prarie , MN	5 %
Robbinsdale, MN	Brooklyn Park, MN	32 %
	Robbinsdale, MN	20 %
	Minneapolis, MN	15 %
Falcon Heights, MN	St. Paul, MN	17 %
	Minneapolis, MN	13 %
	Roseville, MN	10 %

(b) Foursquare preference locality

# Observation: Travel Locality

- Users prefer to travel a limited distance.
  - From Foursquare data analysis:
    - 45% of users travel 10 miles or less
    - 75% of users travel 50 miles or less



# LARS: A Location-Aware Recommender

- LARS: Efficient and scalable Location-aware recommender system that uses location-based ratings.
- Two main considerations/components:
  - Preference locality  $\Leftarrow$  *user partitioning*
    - Collaborative filter utilizing ratings only located in the querying user's region.
  - Travel locality  $\Leftarrow$  *travel penalty*







# Balancing Scalability/Locality

- Challenge: How to balance scalability and locality of partial pyramid?
  - Maintains a large number of regions increases both *locality (higher the better)* and *scalability (lower the better)*.
- Solution: Merging/Split maintenance algorithm
  - $\text{scalability\_gain} < \text{locality\_loss} \Rightarrow$  split the pyramid cell
  - $\text{scalability\_gain} > \text{locality\_loss} \Rightarrow$  merge the pyramid cells



# Travel Penalty for Travel Locality

- Approach:  $RecScore(u, i) = P(u, i) - TravelPenalty(u, i)$
- Challenge: Computational complexity of calculating  $TravelPenalty(u, i)$  for all items online is too high.  $O(k + \log N)$
- Solution: Partition space into grids, and compute the  $TravelPenalty$  of each grid and items offline.



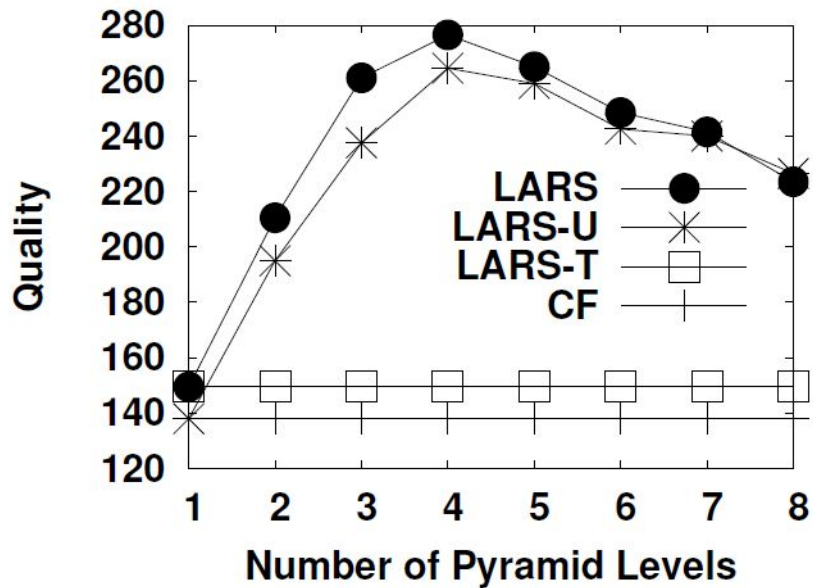
# Datasets for Experiments

- Foursquare data
  - Data crawled from Foursquare application, a location-based SNS.
  - Contains user notes for venues.
- MovieLens data
  - Movie rating data taken from MovieLens.
- Synthetic data
  - Random data for testing scalability and query efficiency.

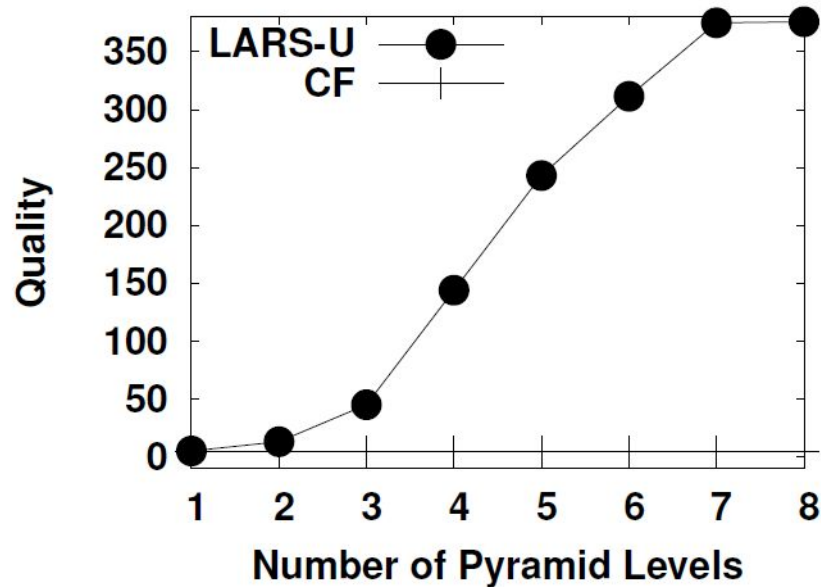


# Evaluation

- LARS surpasses collaborative filter.



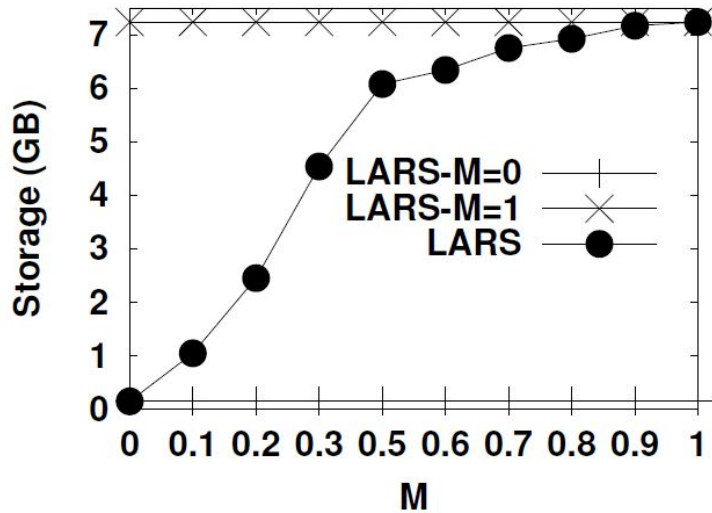
(a) Foursquare data



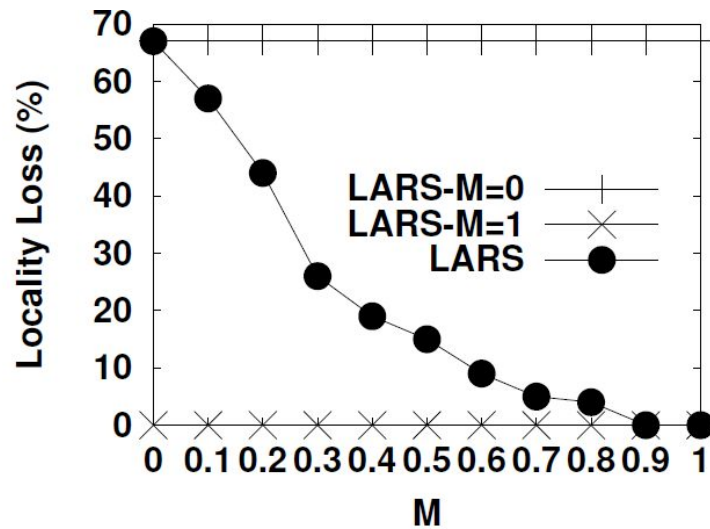
(b) MovieLens data

# Scalability-Locality Tradeoff

- Scalability-locality tradeoff of *partial pyramid*



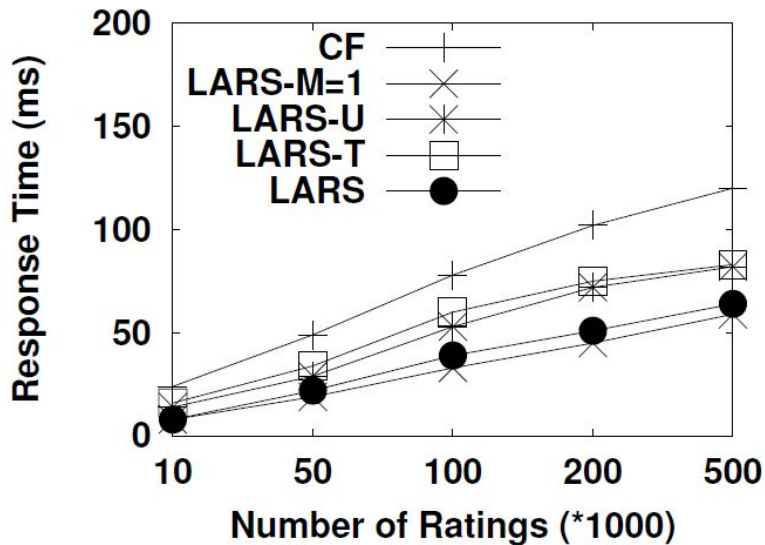
(a) Storage



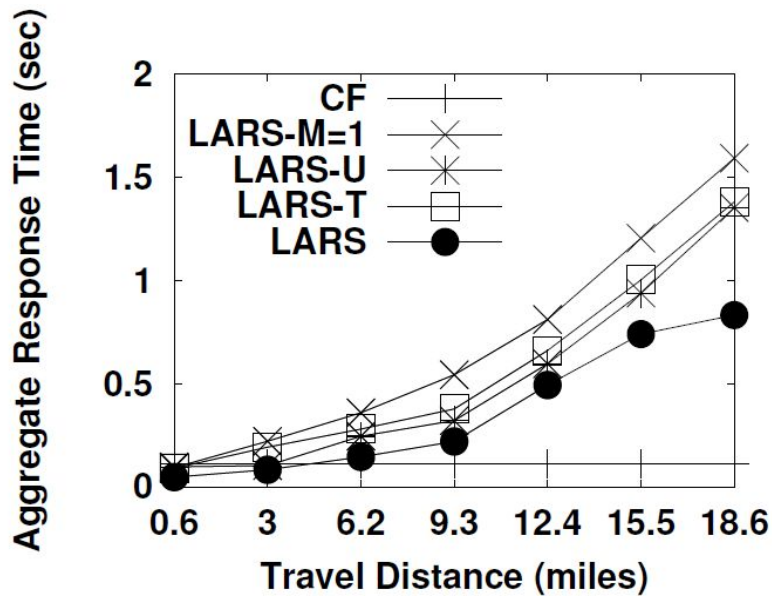
(b) Locality

# Snapshot vs Continuous Queries

- LARS reduces the response time of naive approaches.



(a) Snapshot Queries



(b) Continuous Queries

# Summary

- LARS is the first location-aware recommender system to consider implicit preferences considering user/travel locality.
- LARS effectively leverages computational resources which enables real-world deployment.



# Thank You

## Q & A

