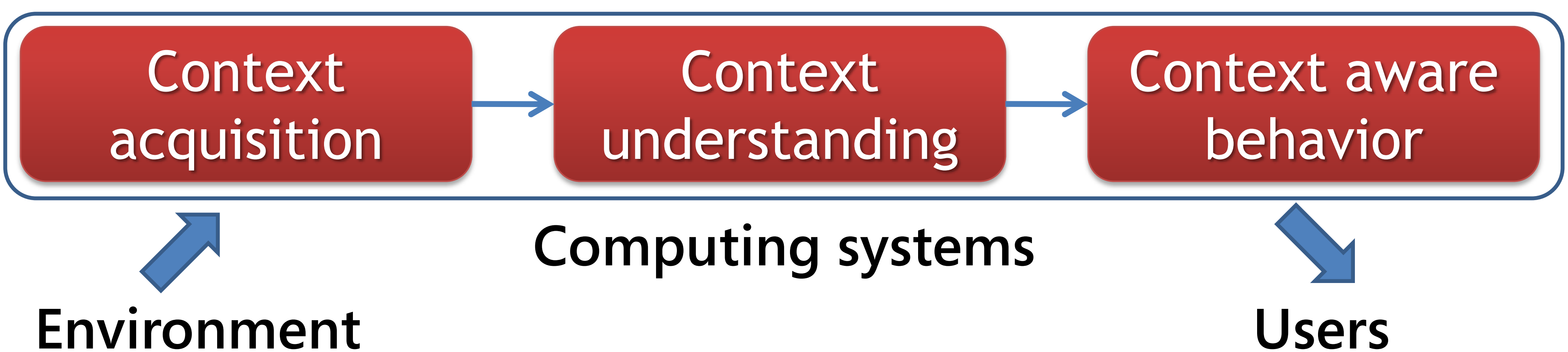
Build Intelligence from the Physical World

Xing Xie
Microsoft Research Asia
Aug. 30, 2011

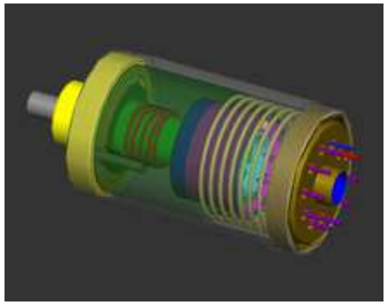
Context Awareness

- A key concept in Ubicomp: deal with linking changes in the environment (physical world) with computing systems
  - Acquisition of context
  - Abstraction and understanding of context
  - Application behavior based on the recognized context
- Build intelligence about physical world in computing systems



Context and Sensors

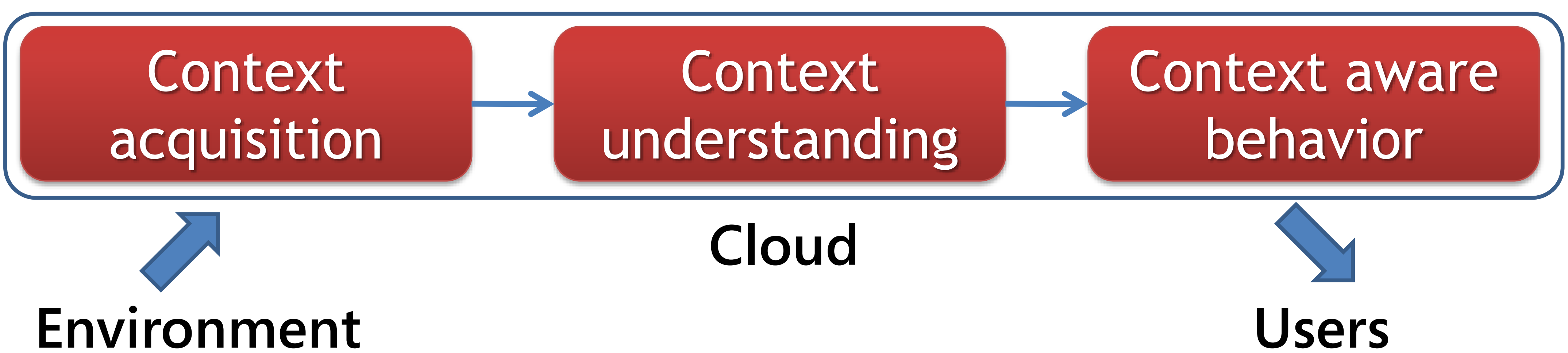
- Sensor: a device that measures a physical quantity and converts it into a signal which can be read by an observer or by an instrument (from Wiki)



- Device time
- Device location
  - GPS, Wi-Fi, cell-tower, Bluetooth
- Device movement
  - Accelerometer, gyroscope
  - Digital compass
- Environment
  - Microphone
  - Camera, ambient light sensor
  - Proximity sensor
  - Barometer, humidity sensor, thermometer

Make the Cloud Intelligent

- The coming era of cloud computing brings new opportunities to this long studied research area
- By accumulating and aggregating context from multiple users, multiple devices, and over a long period, we can obtain collective social intelligence from them



Future Devices = Universal Sensors

7-Layer Architecture

Virtual World

Cloud

- Application layer: search, ad, maps, social, travel, game
- Knowledge layer: user pattern, social pattern, social intelligence
- Security Layer: privacy, trust, identity, policy, permission
- Storage layer: indexing, distributed system, data integrity
- Communication layer: data acquisition, network, protocol

Client

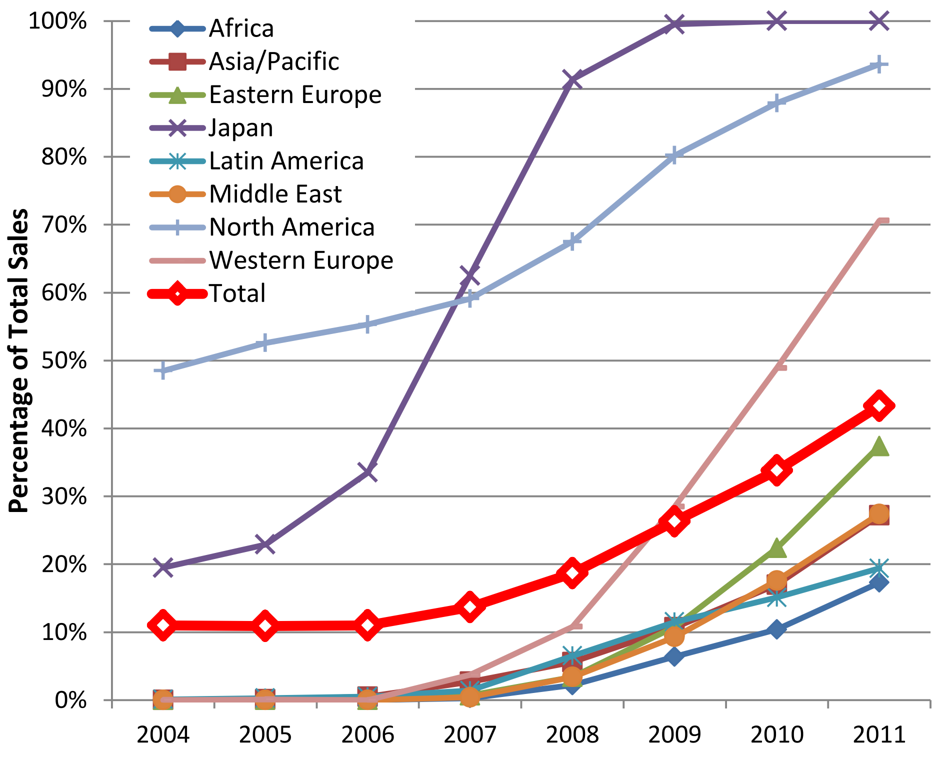
- Communication layer: data acquisition, network, protocol
- Data layer: activity, trajectory, multimedia, profile
- Physical layer: time, location, movement, environment

Physical World

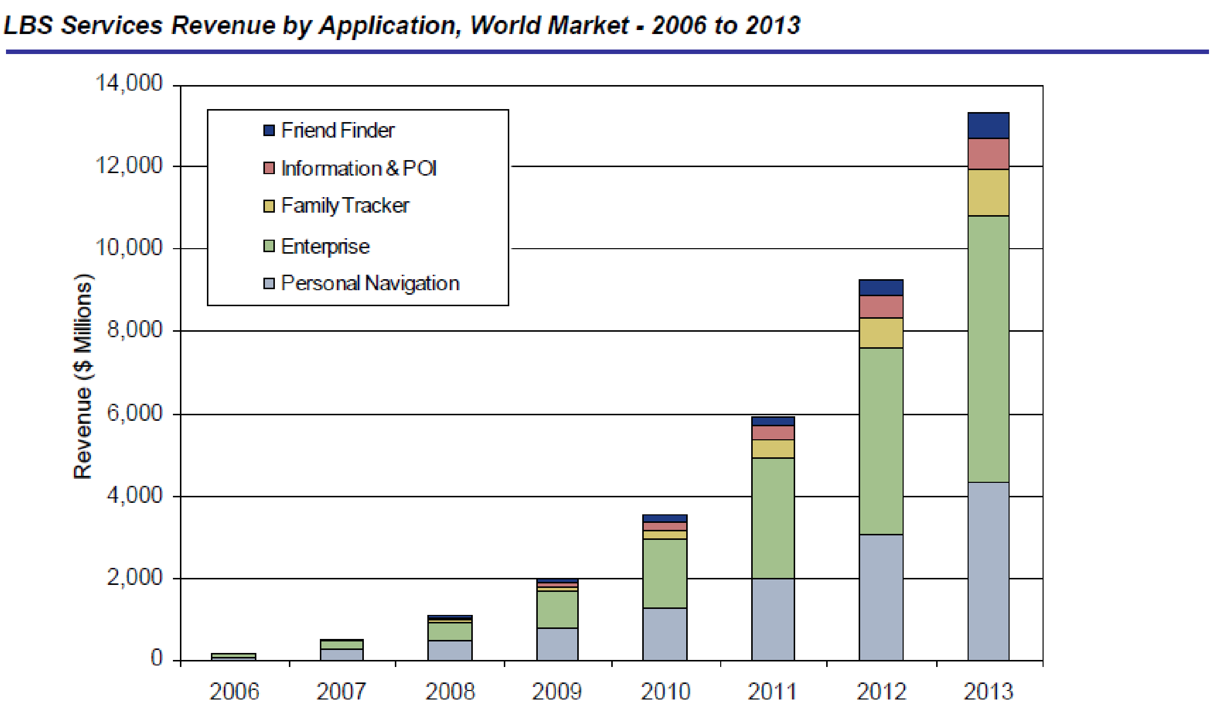
Microsoft Research Asia 10 year anniversary

Location: the Most Important Context Data

- GPS will be installed on 40+% phones by 2011 worldwide
- Location based service (LBS) will become a 13B business by 2013

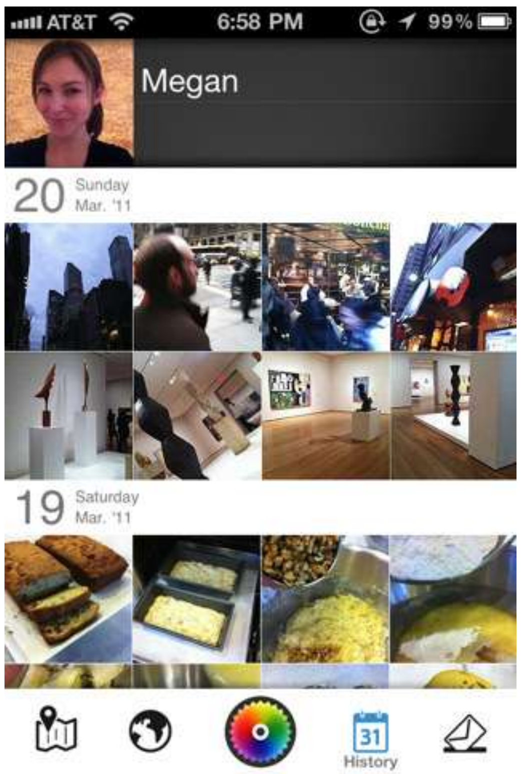


Source: Gartner Dataqueste

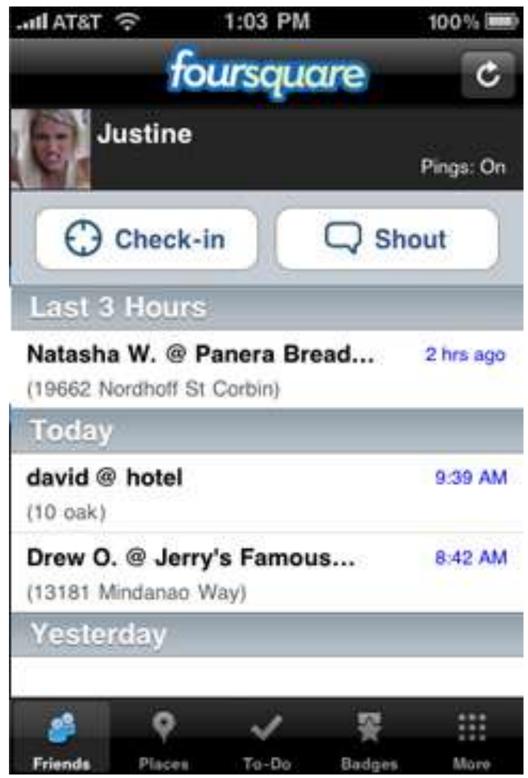


(Source: ABI Research)

Location Based Social Networks



Color



Foursquare

Bedo(贝多)

Projects in MSR Asia

- GeoLife: Building Social Networks Using Human Location History (WWW 2010/2009, AAAI 2010, SIGMOD 2010)

Knowledge from General People → **Social Network Service**

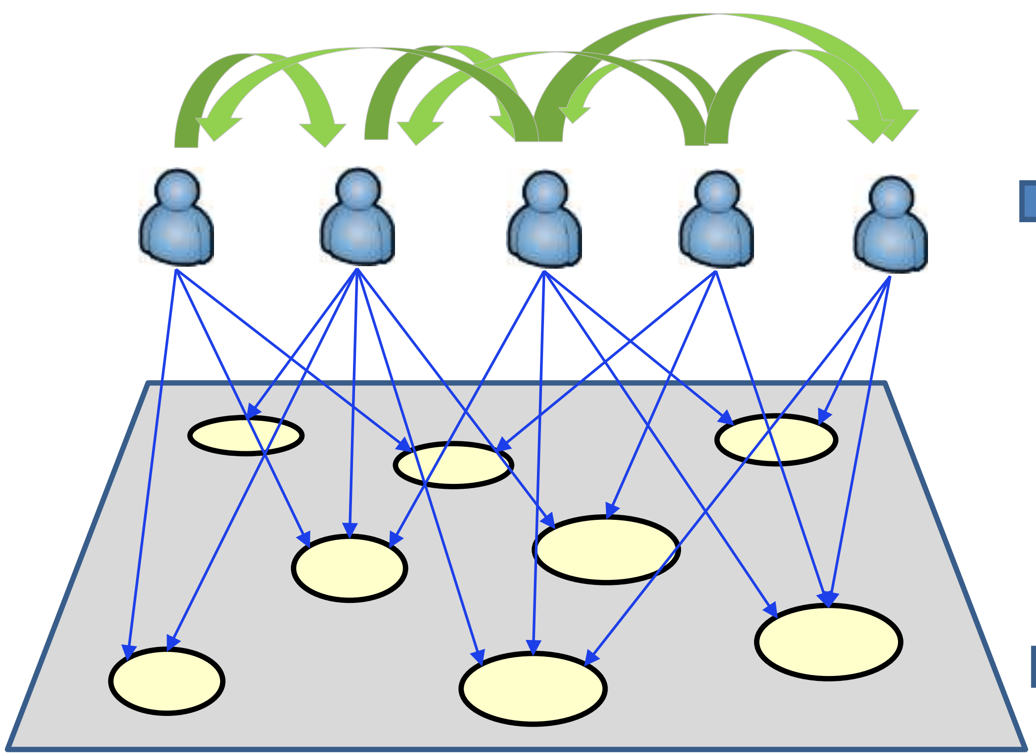
- Mining Geo-Tagged Photos for Travel Recommendation (ACM MM 2010/2009)

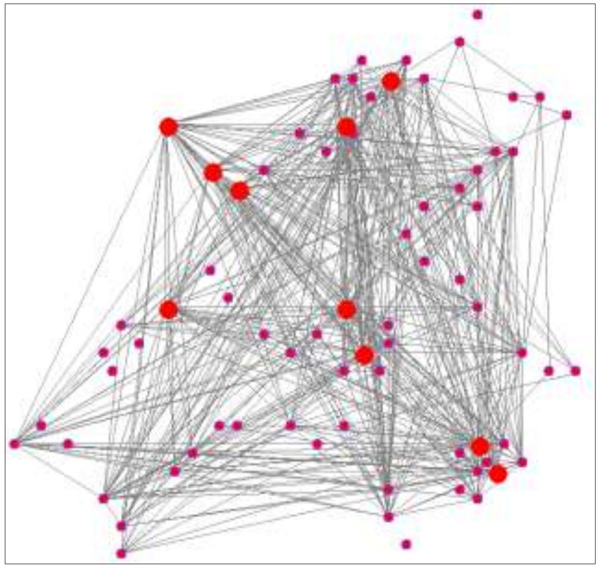
Knowledge from Photographers → **Travel Service**

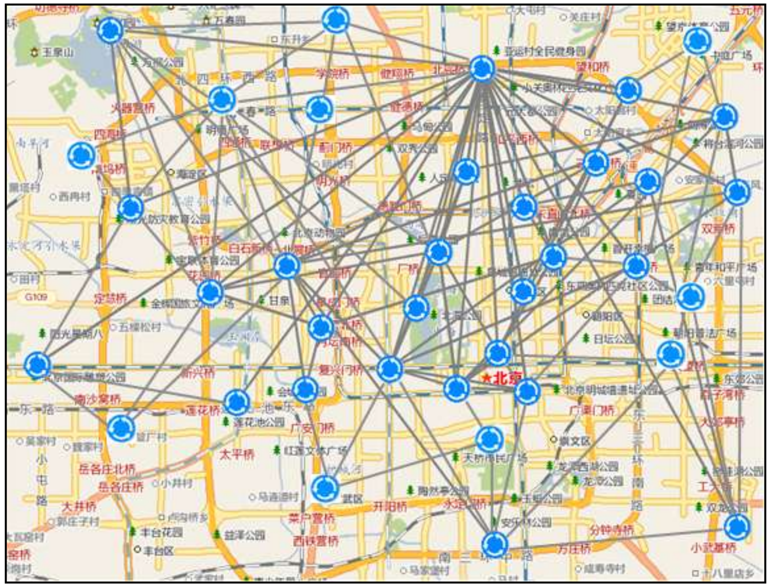
- T-Drive: Driving Directions Based on Taxi Traces (ACM GIS 2010/2009)

Knowledge from Taxi Drivers → **Map and Navigation Service**

GeoLife: Building Social Networks Using Human Location History

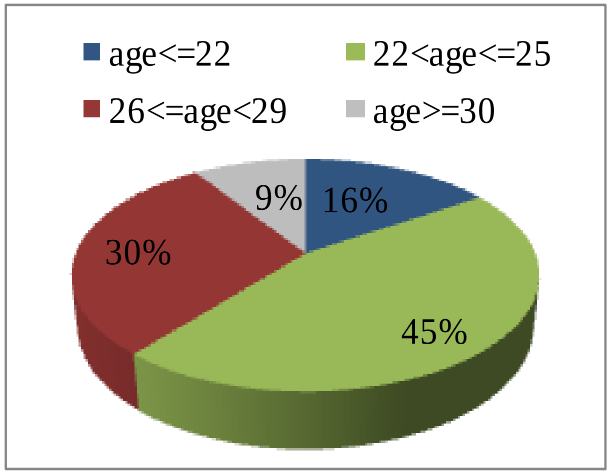


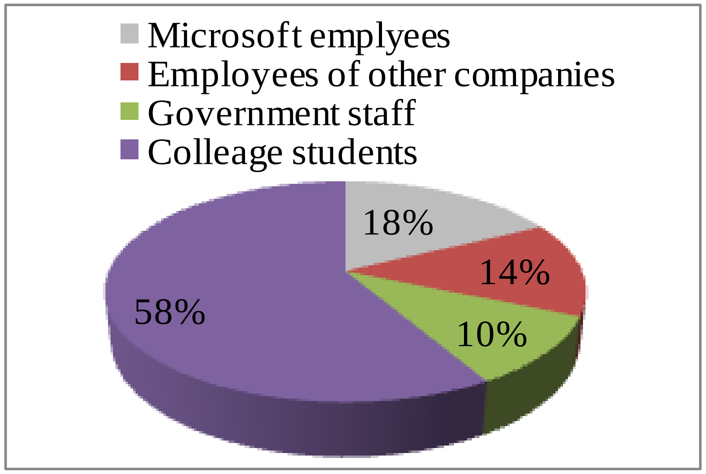




GPS Devices and Users

- 165 users, Apr. 2007 ~ Aug. 2009





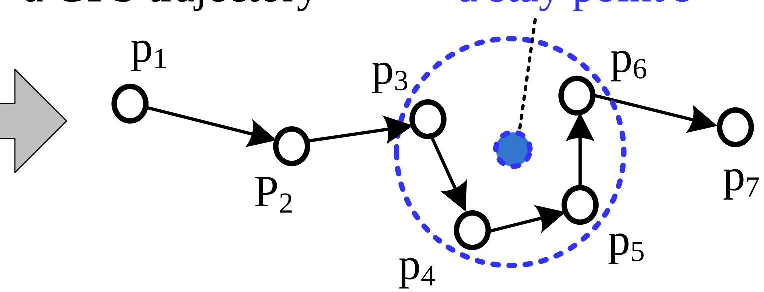
A Free Large-Scale GPS Dataset

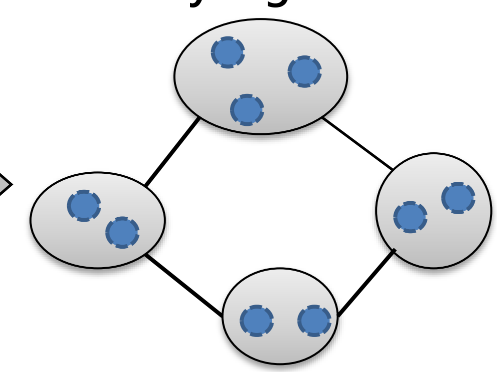
- Shared at my home page (search for “Xing Xie”)
- http://research.microsoft.com/en-us/downloads/b16d359d-d164-469e-9fd4-daa38f2b2e13/default.aspx

GPS Log Processing

- GPS trajectories*

	Latitude,	Longitude,	Arrival Timestamp
p_1:	39.975,	116.331,	9/9/2009 17:54
p_2:	39.978,	116.308,	9/9/2009 18:08
…			
p_K:	39.992,	116.333,	9/12/2009 13:56





Raw GPS points

Stay points

Stay regions

- Stand for a geo-spot where a user has stayed for a while
- Preserve the sequence and vicinity info

- Stand for a geo-region that we may recommend
- Discover the meaningful locations

*** In GPS logs, we have some user comments associated with the trajectories.**

Collaborative Activity and Location Recommendation

- Location Recommendation
  - Question: *I want to find nice food, where should I go?*
- Activity Recommendation
  - Question: *I will visit the downtown, what can I do there?*

Data Modeling

- User <-> Location <-> Activity

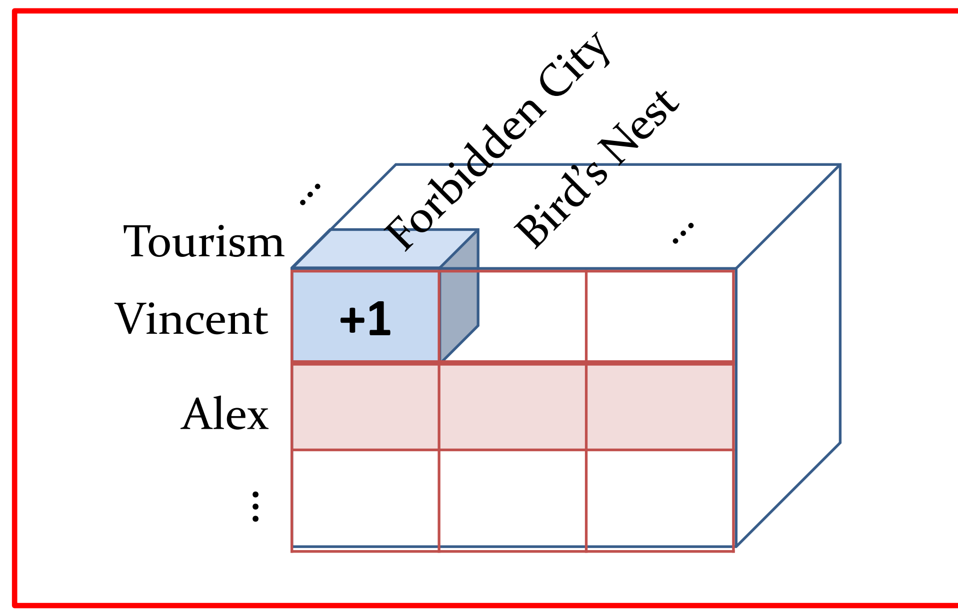
GPS: "39.903, 116.391, 14/9/2009 15:25"

Stay Region: "39.910, 116.400 (Forbidden City)"

"User Vincent: We took a tour bus to see around along the forbidden city moat …"

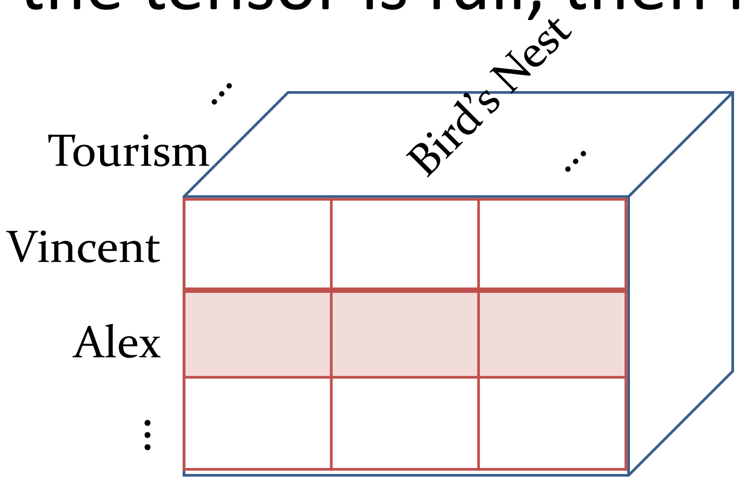
Activity: tourism

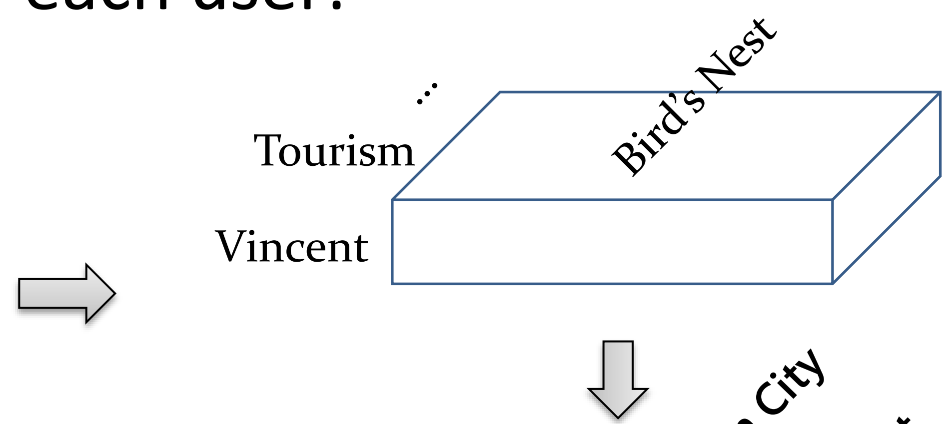
Activities	Descriptions
Food and Drink	Dinning/drinking at restaurants/bars, etc.
Shopping	Supermarkets, department stores, etc.
Movie and Shows	Movie/shows in theaters and exhibition in museums, etc.
Sports and Exercise	Doing exercises at stadiums, parks, etc.
Tourism and Amusement	Tourism, amusement park, etc.



How to Do Recommendation?

- If the tensor is full, then for each user:





	Forbidden City	Bird's Nest	Zhongguancun
Shopping	2	1	6
Exhibition	4	3	2
Tourism	5	4	1

Location recommendation for Vincent
Tourism:
Forbidden City > Bird's Nest > Zhongguancun

Activity recommendation for Vincent
Forbidden City:
Tourism > Exhibition > Shopping

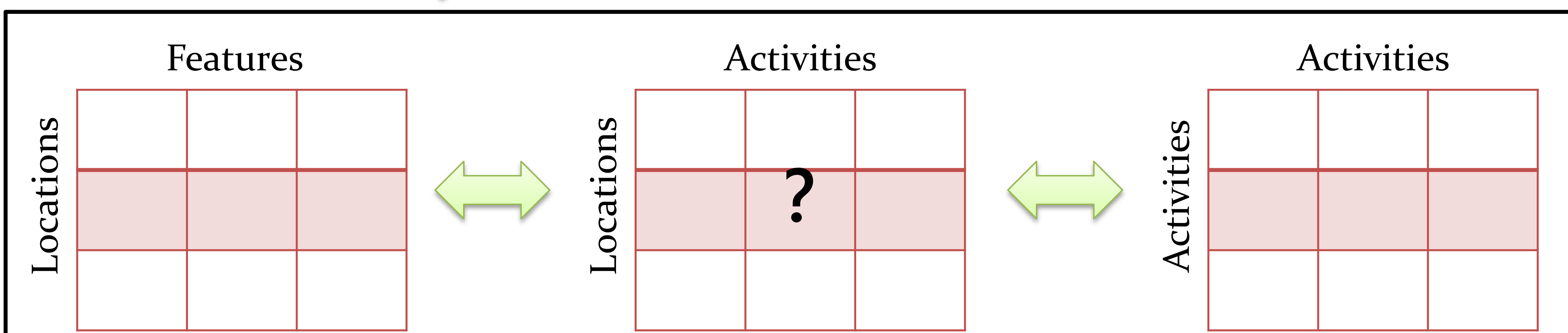
Unfortunately, in practice, the tensor is usually sparse!

Our First Solution (WWW 2010)

	Tourism	Exhibition	Shopping
Forbidden City	5	?	?
Bird's Nest	?	1	?
Zhongguancun	1	?	6

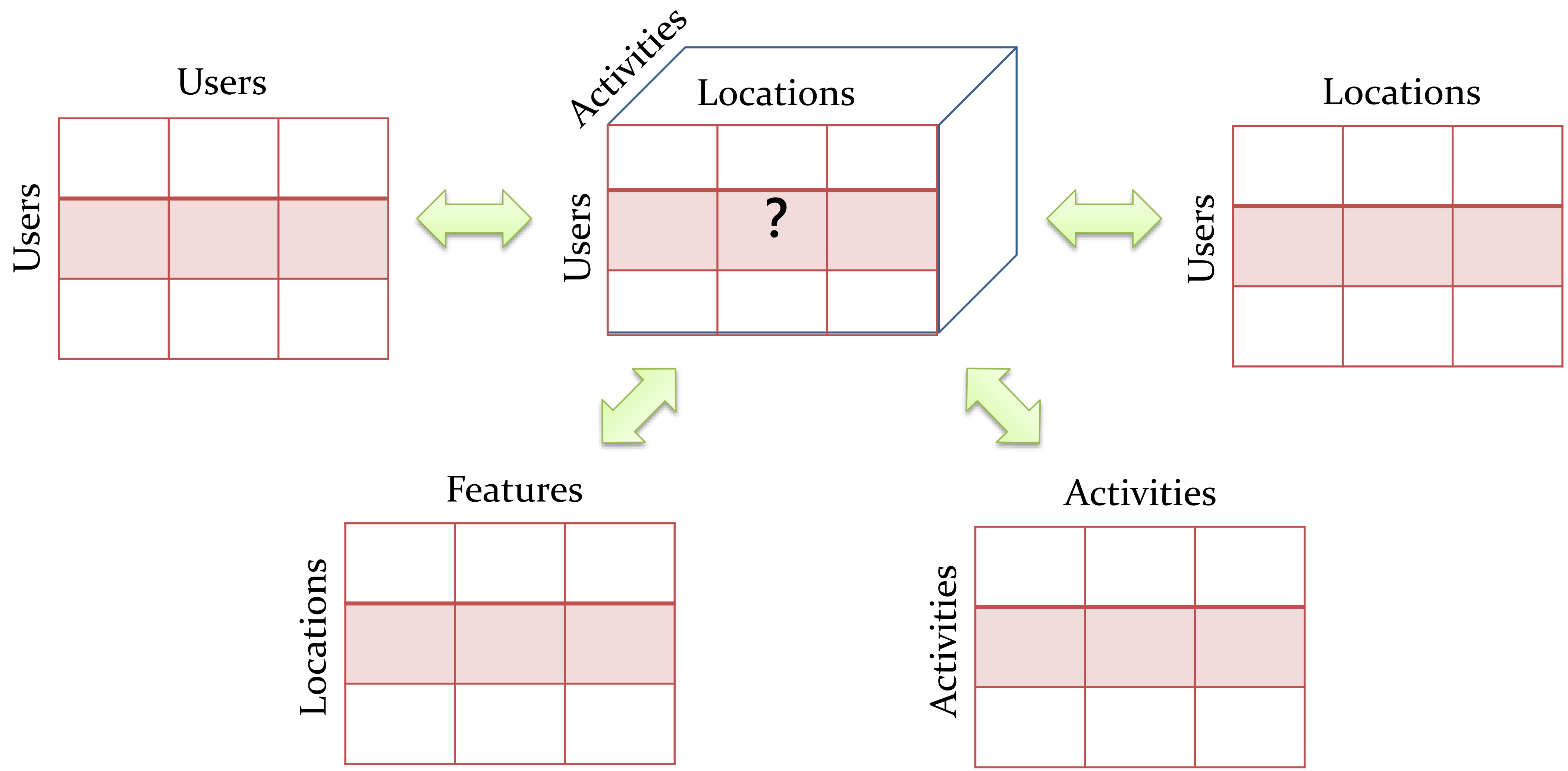
User not explicitly modeled!

1. Not modeling each single user's Loc-Act history
2. = a sum compression of our tensor

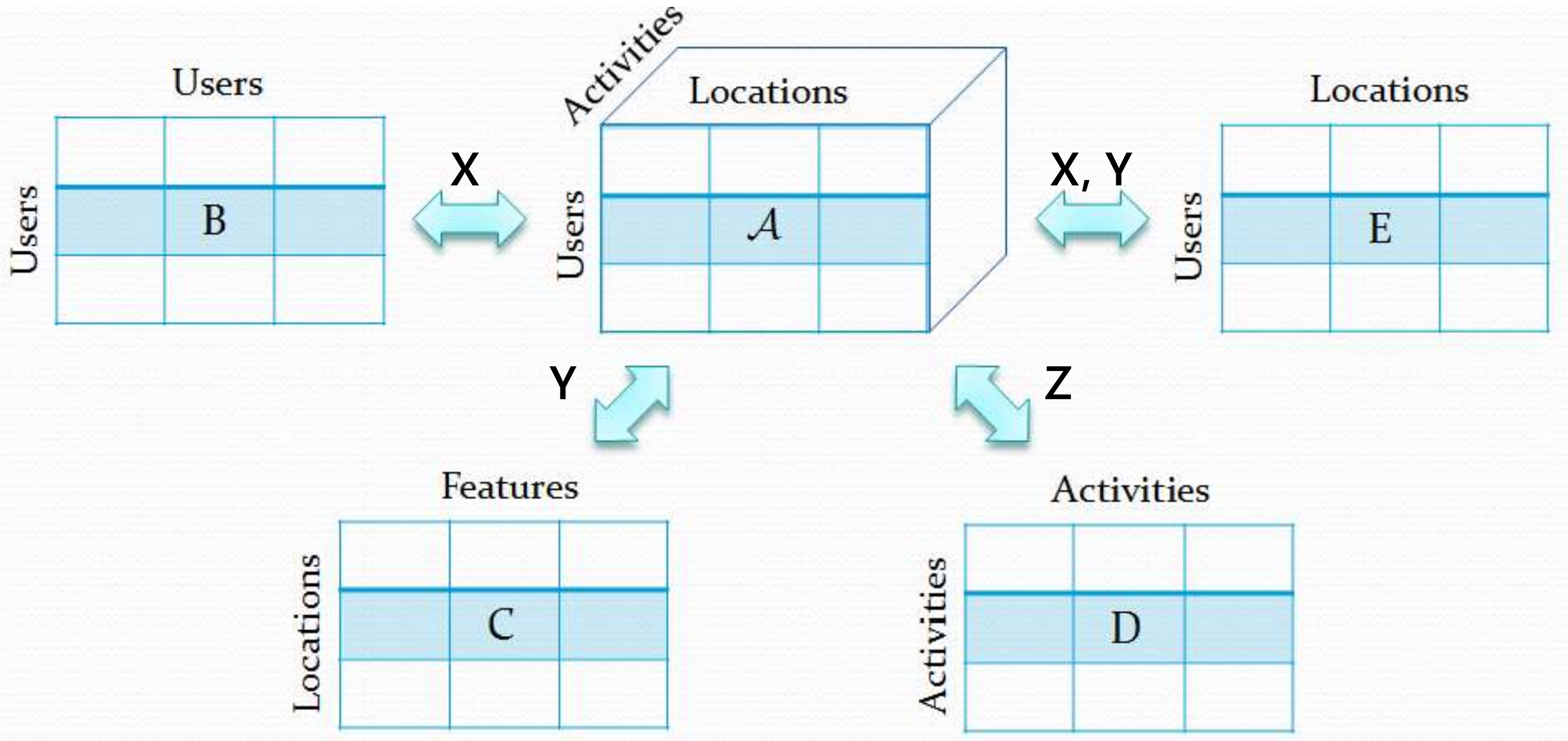


Our Second Solution

- Regularized Tensor and Matrix Decomposition



Our Model



$$
\begin{aligned}
\mathcal{L}(X,Y,Z,U) = \tfrac{1}{2}\left\|\mathcal{A} - [\![X,Y,Z]\!]\right\|^2 \\
+ \tfrac{\lambda_1}{2}\mathrm{tr}(X^T L_B X) + \tfrac{\lambda_2}{2}\left\|C - YU^T\right\|^2 + \tfrac{\lambda_3}{2}\mathrm{tr}(Z^T L_D Z) + \tfrac{\lambda_4}{2}\left\|E - XY^T\right\|^2 \\
+ \tfrac{\lambda_5}{2}(\|X\|^2 + \|Y\|^2 + \|Z\|^2 + \|U\|^2)
\end{aligned}
$$

Location Feature Extraction

- Location features: Points of Interests (POIs)

Stay Region: “39.980, 116.306 (Zhongguancun)”

[restaurant, bank, shop] = [3, 1, 1]

TF-IDF style normalization*: feature = [0.13, 0.32, 0.18]

	restaurant	bank	…
Forbidden City			
Zhongguancun	0.13	0.32	
⋮			

Location-Feature Matrix

TF-IDF (Term-Frequency Inverse Document Frequency):

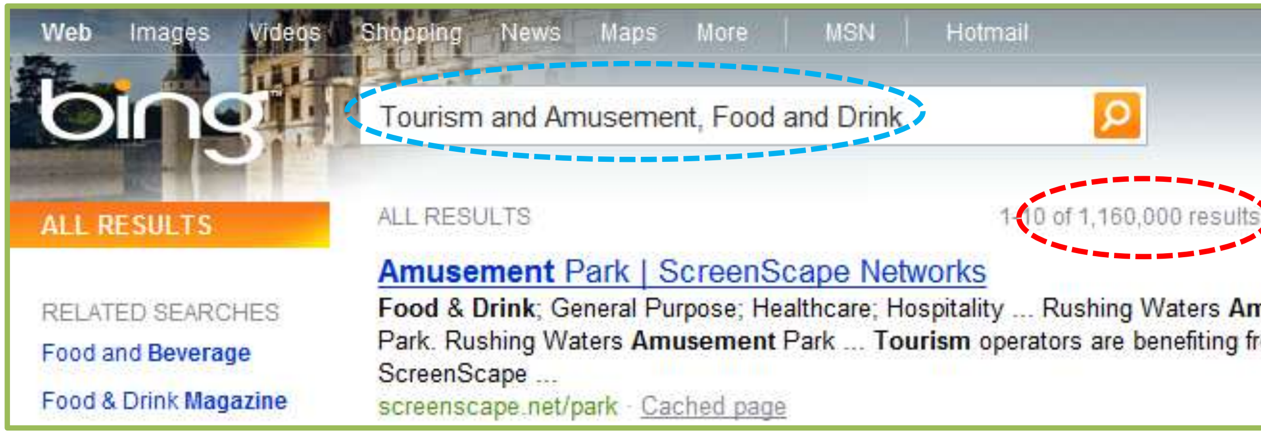
$$tf\text{-}idf_{i,t} = \frac{n_{i,t}}{\sum_l n_{i,l}} \cdot \log \frac{|\{d_i\}|}{|\{d_i : t \in d_i\}|}$$

Example:
Assume in 10 locations, 8 have restaurants (less distinguishing), while 2 have banks and 4 have shops:

tf-idf(restaurant) = (3/5)*log(10/8) = 0.13
tf-idf(bank) = (1/5)*log(10/2) = 0.32
tf-idf(shop) = (1/5)*log(10/4) = 0.18

Activity Correlation Extraction

- How possible for one activity to happen, if another activity happens?
  - Automatically mined from the Web, potentially useful when #(act) is large

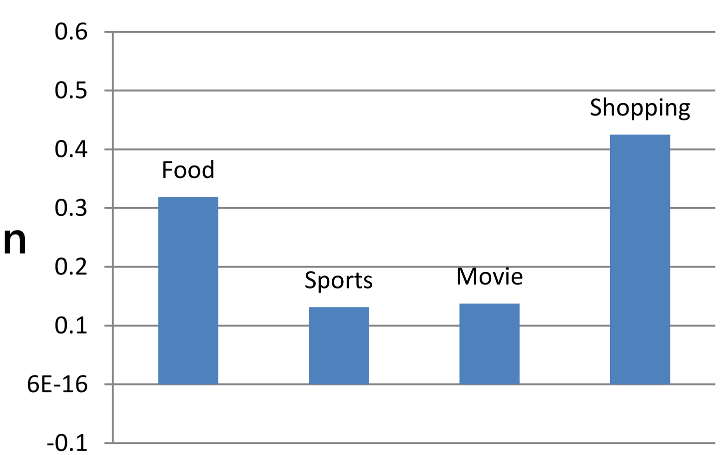


"Tourism and Amusement"
and
"Food and Drink"

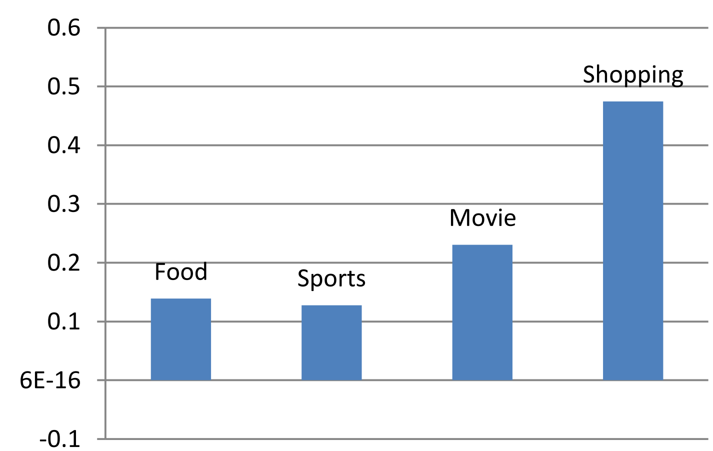
Correlation = h(1.16M),
where h is a normalization func.

Most mined correlations are reasonable. Example: "Tourism" with other activities.

Tourism-Shopping more likely to happen together than *Tourism-Sports*



Web search (from Bing)



Human design (average on 8 subjects)

Optimization

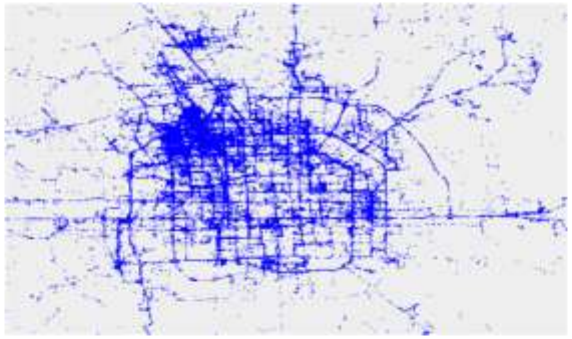
- Minimize the object function *L*(X, Y, Z, U)
  - Gradient descent

$$X_{t+1} = X_t - \gamma \nabla_X,\ Y_{t+1} = Y_t - \gamma \nabla_Y,\ Z_{t+1} = Z_t - \gamma \nabla_Z, U_{t+1} = U_t - \gamma \nabla_U$$

where
$$\begin{aligned}
\nabla_X \mathcal{L} &= -A^{(1)}(Z * Y) + X\left[(Z^T Z) \odot (Y^T Y)\right] \\
&\quad + \lambda_1 L_B X + \lambda_4 (XY^T - E)Y + \lambda_5 X, \\
\nabla_Y \mathcal{L} &= -A^{(2)}(Z * X) + Y\left[(Z^T Z) \odot (X^T X)\right] \\
&\quad + \lambda_2 (YU^T - C)U + \lambda_4 (XY^T - E)^T X + \lambda_5 Y, \\
\nabla_Z \mathcal{L} &= -A^{(3)}(Y * X) + Z\left[(Y^T Y) \odot (X^T X)\right] \\
&\quad + \lambda_3 L_D Z + \lambda_5 Z, \\
\nabla_U \mathcal{L} &= \lambda_2 (YU^T - C)^T Y + \lambda_5 U,
\end{aligned}$$

  - Complexity: O (T × (mnr + m^2 + r^2))
    - T is #(iteration), m is #(user), n is #(location), r is #(activity)

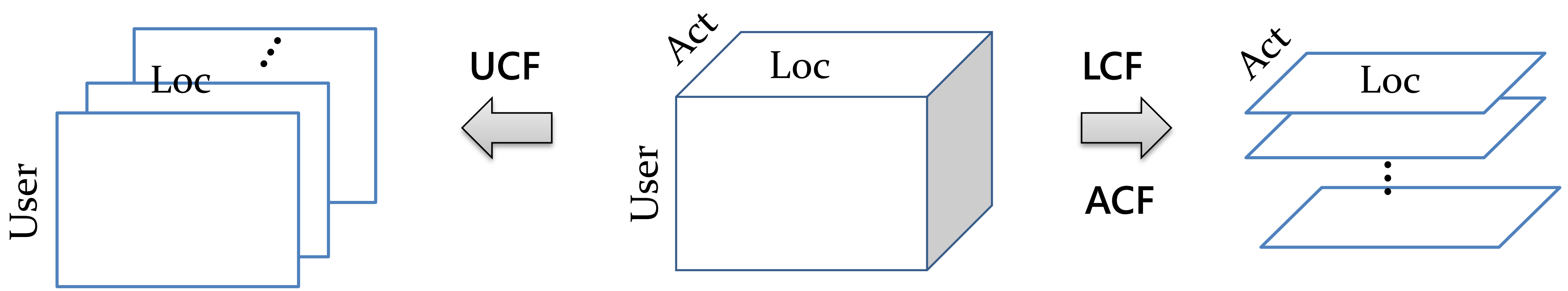
Experiments



- Data
  - GeoLife data set
  - 13K GPS trajectories, 140K km long
  - 530 comments
  - After clustering, #(loc) = 168; #(user) = 164, #(act) = 5, #(loc_fea) = 14
  - The user-loc-act tensor has 1.04% of the entries with values
- Evaluation
  - Ranking over the hold-out test dataset
  - Metrics:
    - Root Mean Square Error (RMSE)
    - Normalized discounted cumulative gain (*nDCG*)

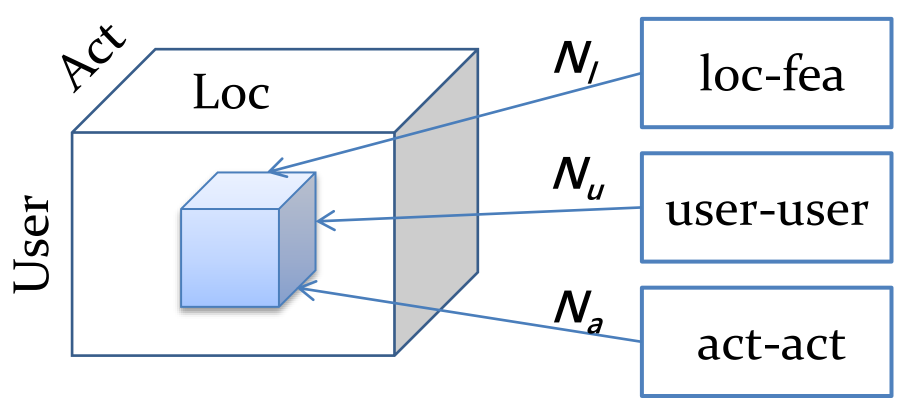
Baselines – Category I

- Tensor -> Independent matrices [Herlocker et al. 1999]
  - Baseline 1: UCF (user-based CF)
    - CF on each user-loc matrix + Top *N* similar users for weighted average
  - Baseline 2: LCF (location-based CF)
    - CF on each loc-act matrix + Top *N* similar locations for weighted average
  - Baseline 3: ACF (activity-based CF)
    - CF on each loc-act matrix + Top *N* similar activities for weighted average

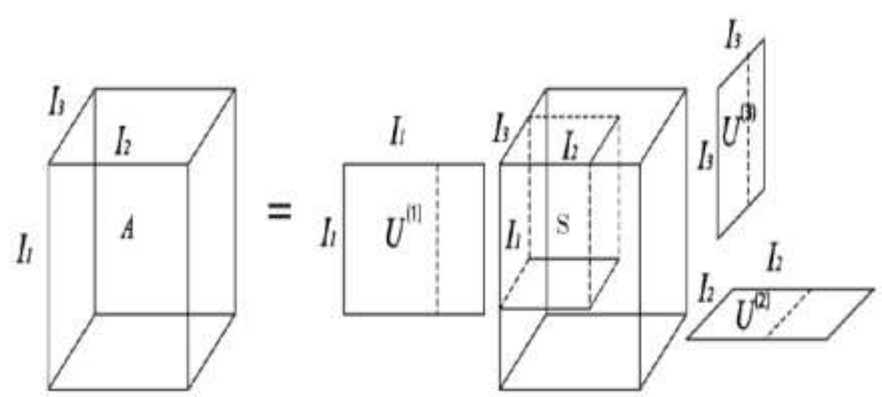


Baselines – Category II

- Tensor-based CF
  - Baseline 4: ULA (unifying user-loc-act CF) [Wang et al. 2006]
    - Top N_u similar users, top N_l similar loc's, top N_a similar act's
    - Similarities from additional matrices + Small cube for weight average
  - Baseline 5: HOSVD (high order SVD) [Symeonidis et al. 2008]
    - Singular value decomposition with matrix unfolding



ULA

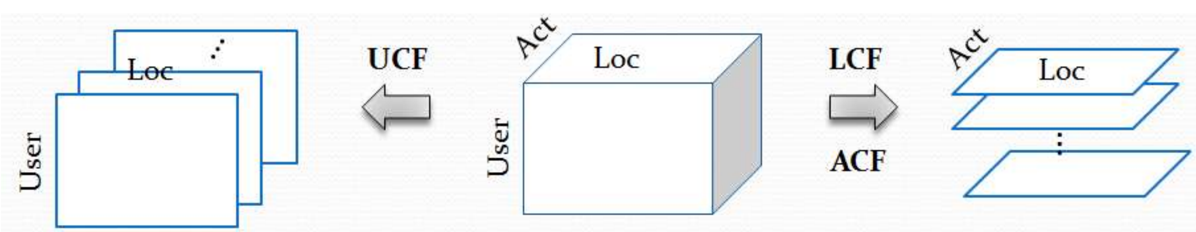


HOSVD

Comparison with Baselines

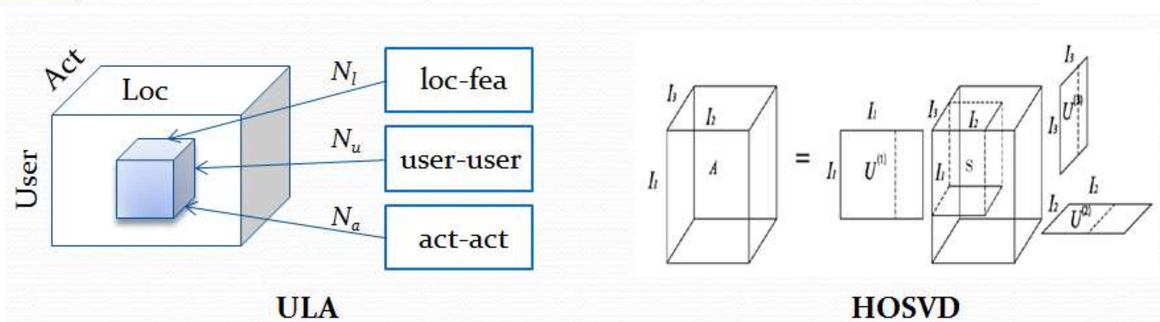
- Reported in “mean ± std”

	RMSE	nDCG_{loc}	nDCG_{act}
UCF	0.027 ± 0.006	0.297 ± 0.024	0.807 ± 0.007
LCF	0.009 ± 0.000	0.532 ± 0.021	0.614 ± 0.019
ACF	0.022 ± 0.005	0.408 ± 0.012	0.785 ± 0.006
ULA	0.015 ± 0.003	0.291 ± 0.022	0.799 ± 0.012
HOSVD	0.006 ± 0.001	0.390 ± 0.021	0.913 ± 0.004
UCLAF	$\mathbf{0.006 \pm 0.001}$	$\mathbf{0.599 \pm 0.036}$	$\mathbf{0.959 \pm 0.009}$



[Herlocker et al. 1999]

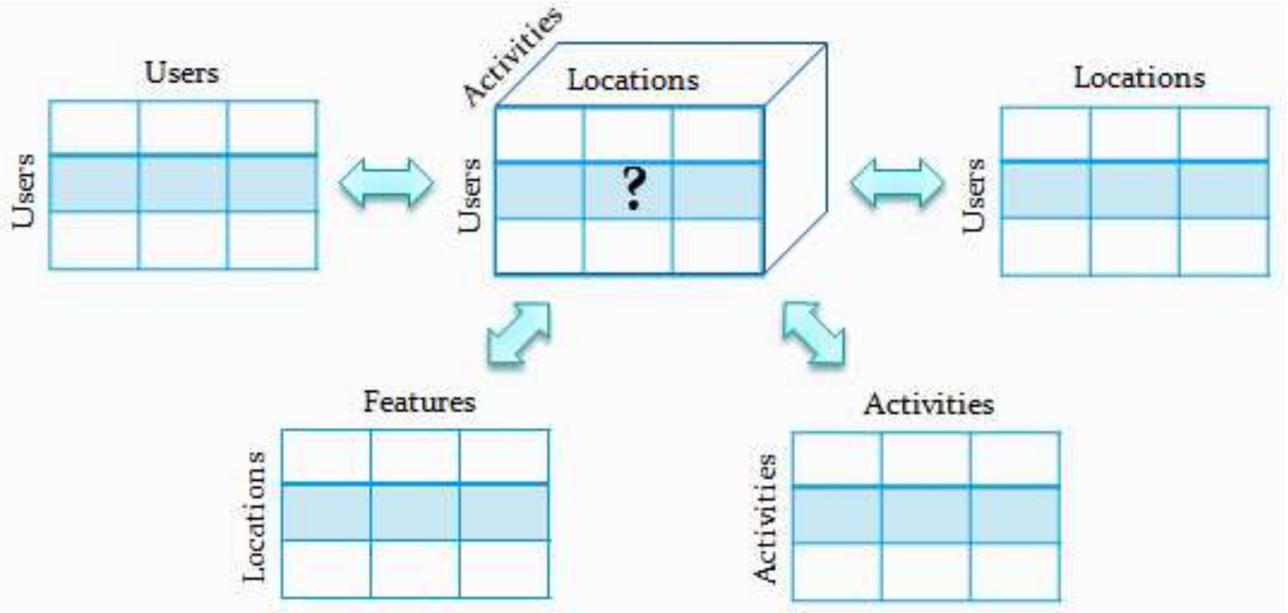
[Wang et al. 2006]



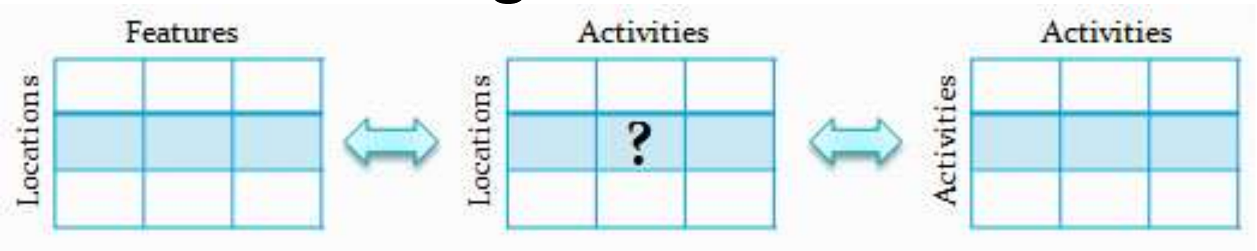
[Symeonidis et al. 2008]

Comparison with Our First Solution

- Current user-centric solution



- Previous generic solution

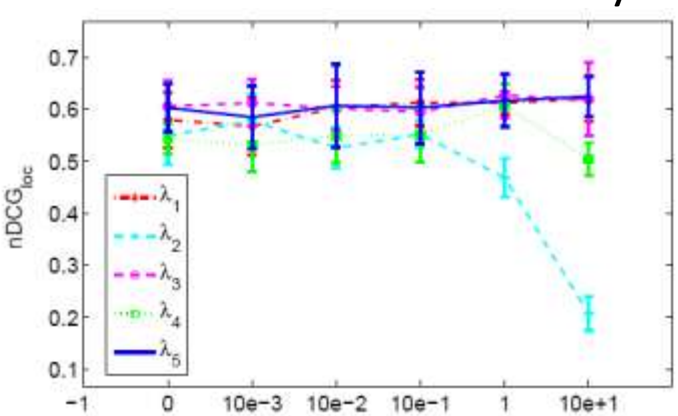


Performance

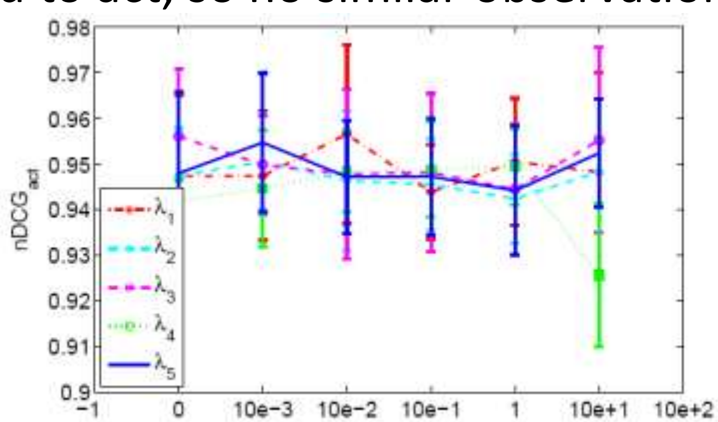
	Current Solution	Previous Solution
RMSE	**0.006 ±0.001**	0.041 ±0.006
$nDCG_{loc}$	**0.576 ±0.043**	0.552 ±0.027
$nDCG_{act}$	**0.931 ±0.009**	0.885 ±0.019

Impacts of the Model Parameters

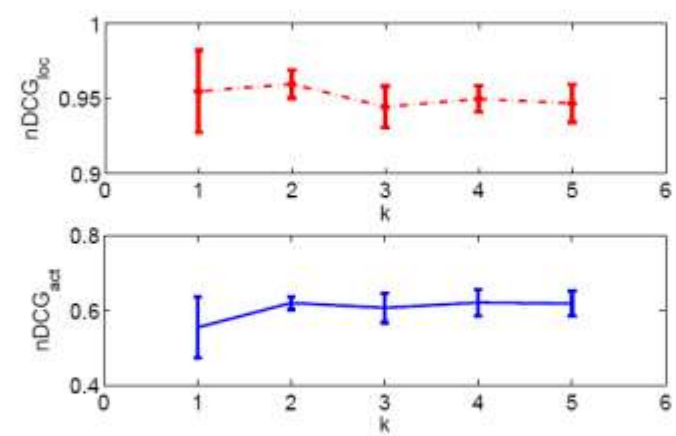
- Some observations
  - Using additional info (i.e. $\lambda_i > 0$) is better than not (i.e. $\lambda_i = 0$)
  - Not very sensitive to most parameters
    - Model is robust + Contribution from additional info is limited
  - As λ_2 increases, nDCG for loc recommendation greatly decreases
    - Maybe because the loc-feature matrix is noisy in extracting the POIs
    - Not directly related to act, so no similar observation for act recommendation



(a) Impact of λ_i's to location recommend.



(b) Impact of λ_i's to activity recommend.



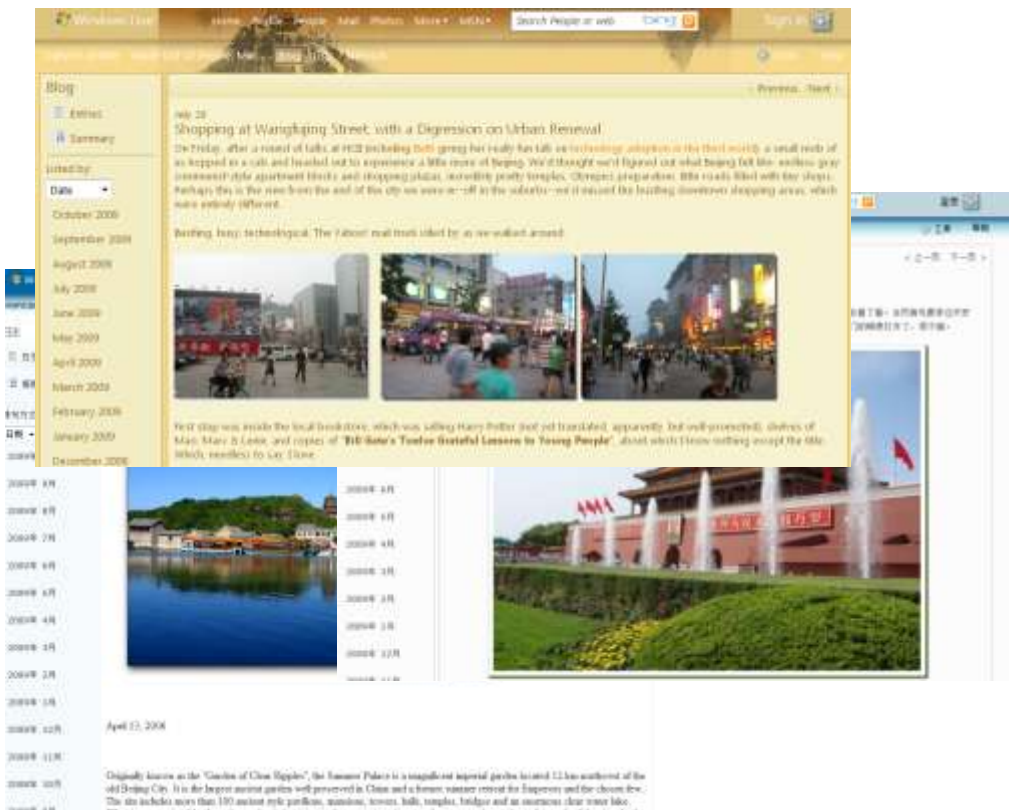
(c) Impact of the low dimension k

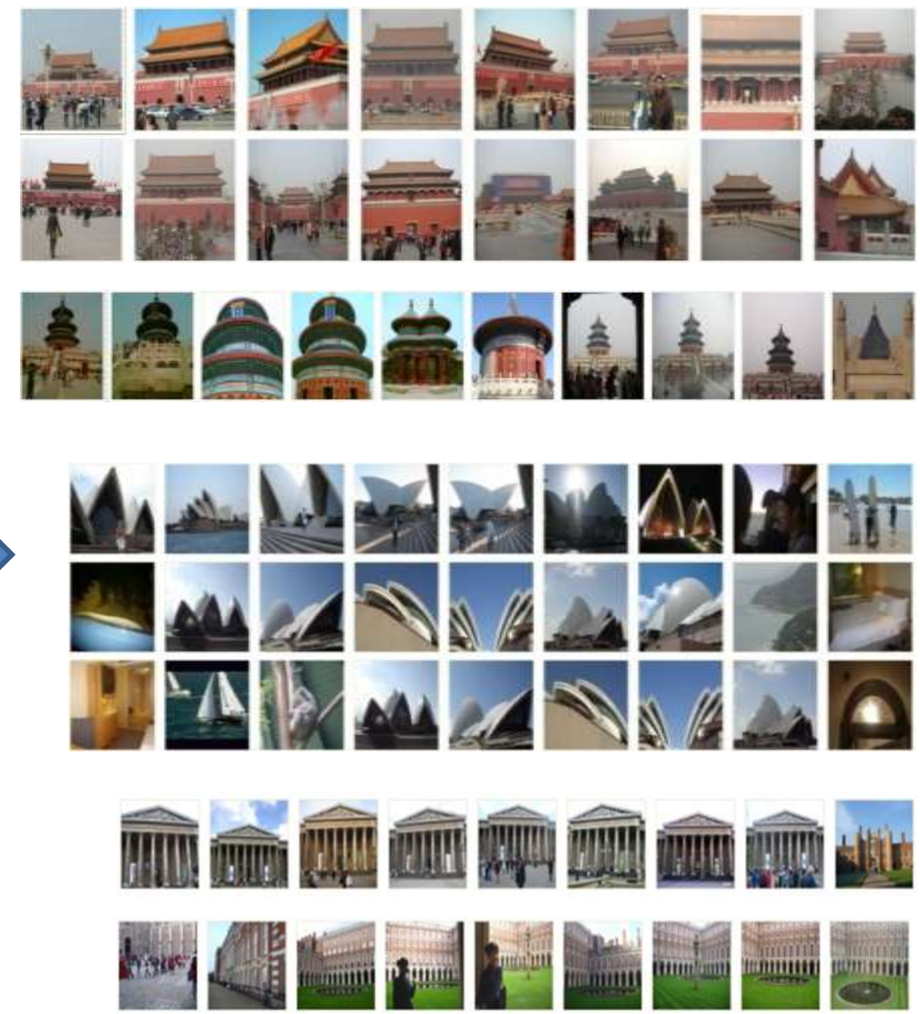
Collaborative Activity and Location Recommendation

- We showed how to mine knowledge from GPS data to answer
  - If I want to do something, where should I go?
  - If I will visit some place, what can I do there?
- We evaluated our system on a large GPS dataset
  - 19% improvement on location recommendation
  - 22% improvement on activity recommendation

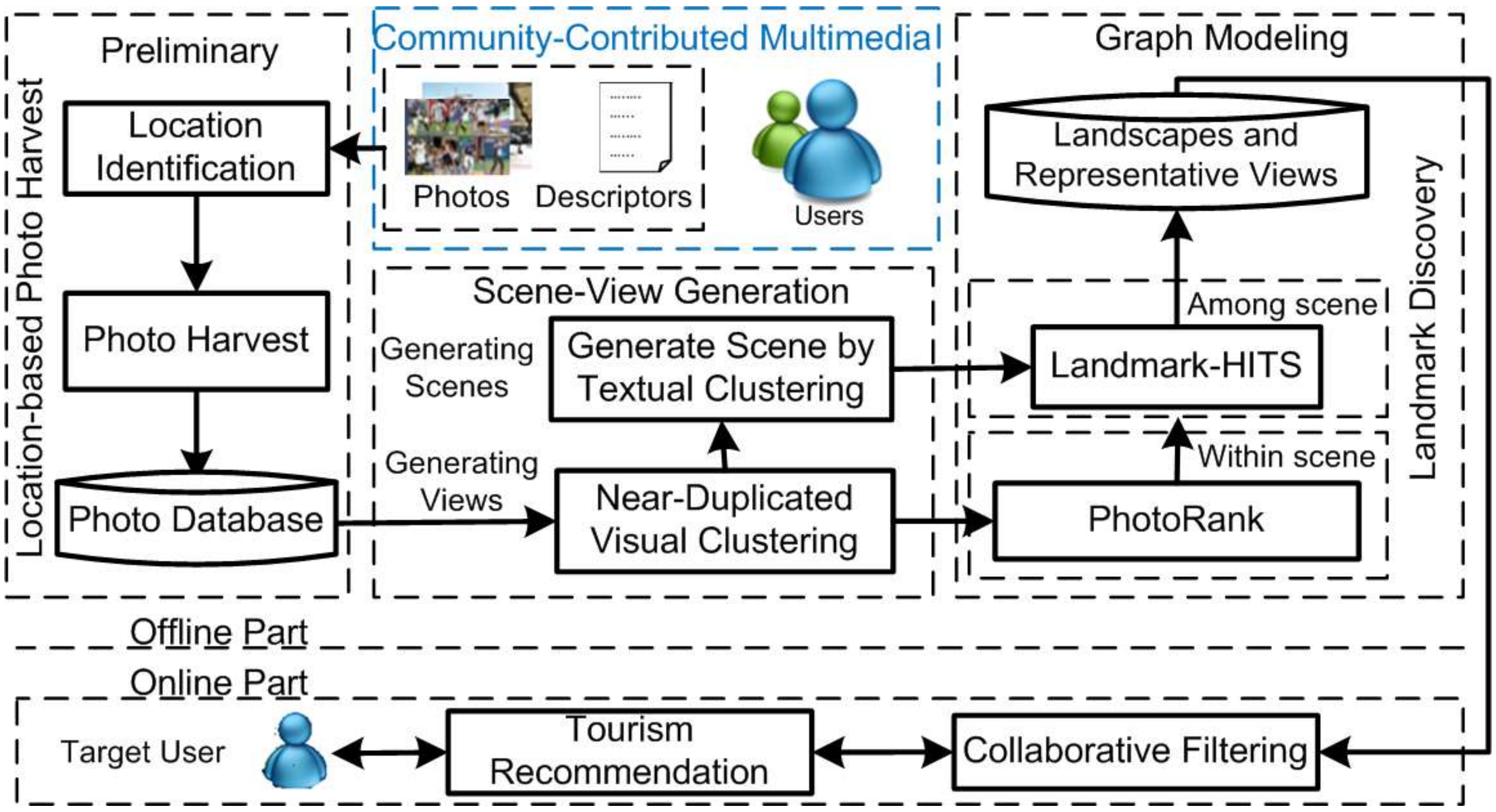
  over the simple memory-based CF baseline (i.e. UCF, LCF, ACF)

Mining City Landmarks from Photos

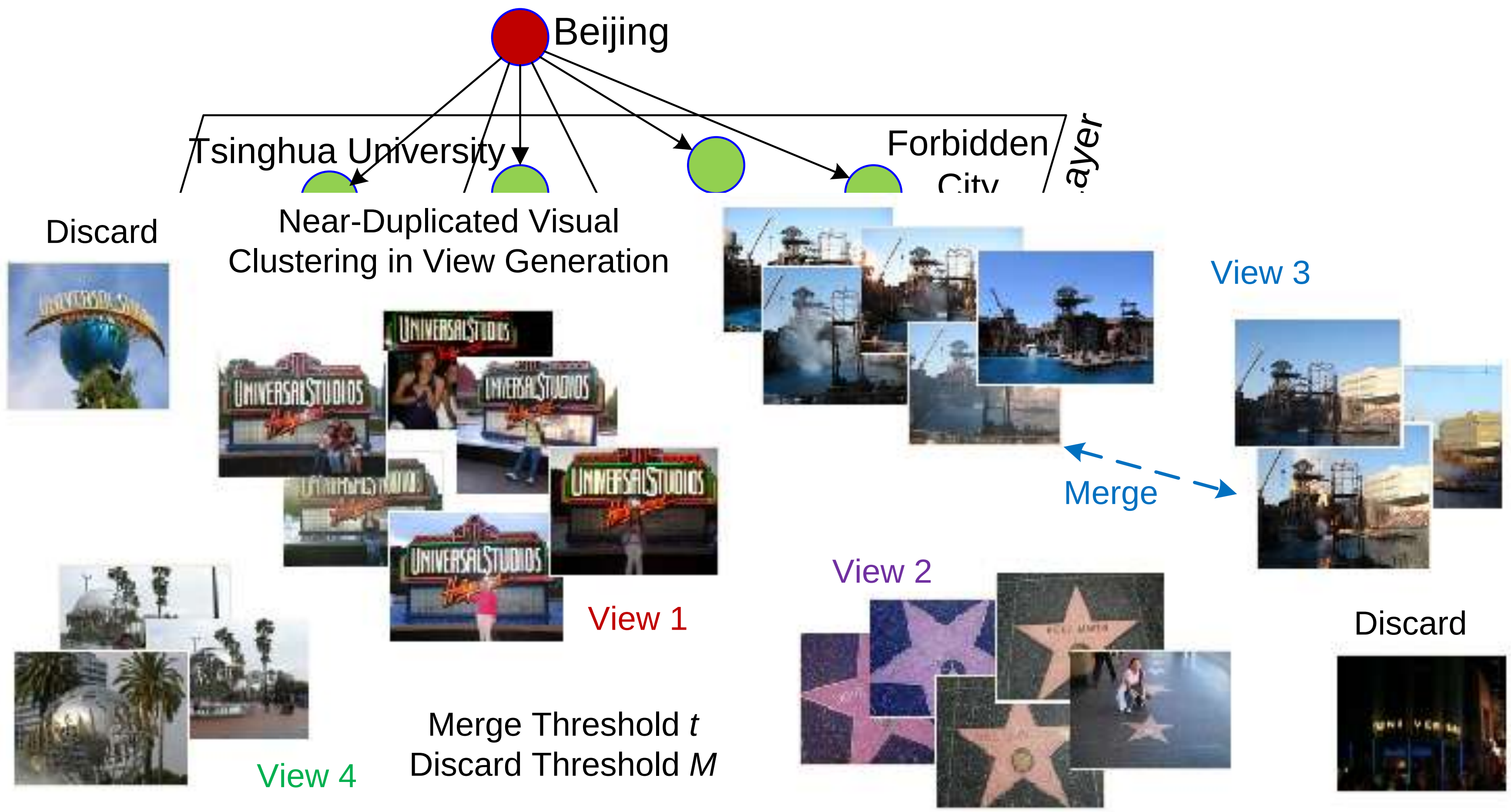




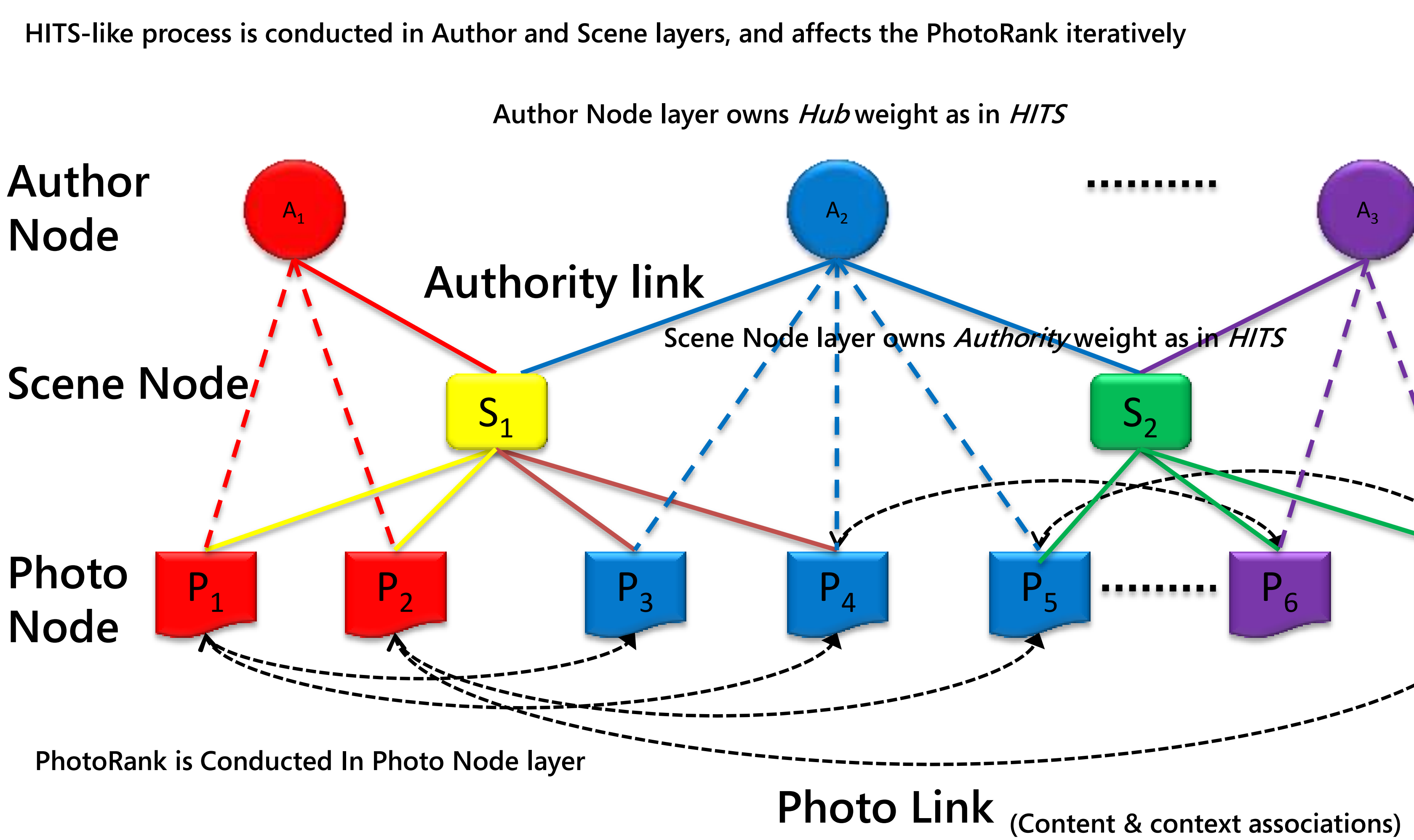
System Framework



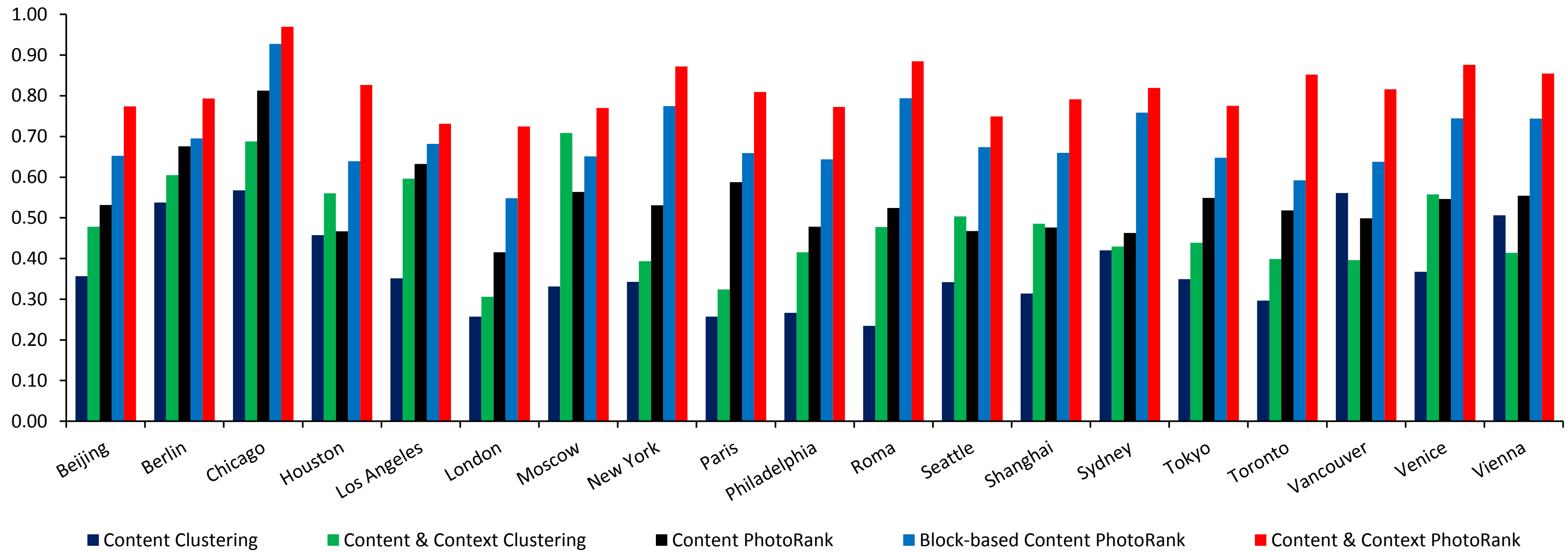
View Generation by Visual Clustering



Mining Landmarks by Graph Modeling



Experimental Results



Blog Users	Top Ranked Landmarks at Worldwide Scale by Landmark-HITS
Asian	1. Summer Palace (Beijing), 2. Sydney Opera House (Sydney), 3. Louvre Museum (Paris), 4. Tiananmen (Beijing), 5. Tokyo Tower (Tokyo), 6. Universal studios (L.A.), 7. Oriental Pearl (Shanghai), 8. Tower of London (London), 9. Empire State Building (New York), 10. Statue of Liberty (New York)
European	1. Sydney Opera House (Sydney), 2. Louvre Museum (Paris), 3. London Museum (London), 4. Summer Palace (Beijing), 5. Tower of London (London), 6. Empire State Building (New York), 7. Statue of Liberty (New York), 8. Oriental Pearl (Shanghai), 9. Tokyo Tower (Tokyo), 10. Universal studios (L.A.)
American	1. Statue of Liberty (New York), 2. Universal studios (L.A.), 3. Sydney Opera House (Sydney), 4. Empire State Building (New York), 5. Louvre Museum (Paris), 6. Space Needle (Seattle), 7. Summer Palace (Beijing), 8. Cn Tower (Toronto), 9. Tokyo Tower (Tokyo), 10. Oriental Pearl (Shanghai)

Mining Trip Knowledge from Geo-tagged Photos

- Trace people's trips from geo-tagged photo collections
- Photo trip patterns:
  - Sequence of visited cities and durations of stay
  - Typical description of trips represented by tags
- Classify photo trip patterns based on their trip themes

Milano $\xrightarrow{\text{1 day}}$ Venice $\xrightarrow{\text{2 days}}$ Pisa

Milano	Venice	Pisa
•Duomo •La Scala	•Gondola •San Marco	•Leaning Tower

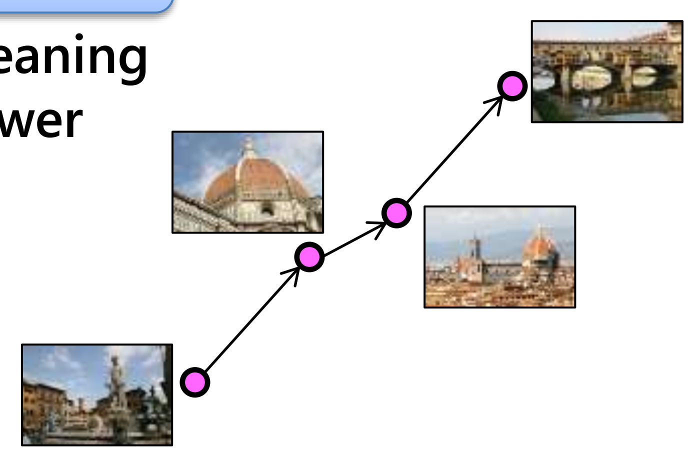


Photo Trip Pattern Mining: Segmentation

- Detect changes of trips based on captured time gaps, distance between photos, and tags

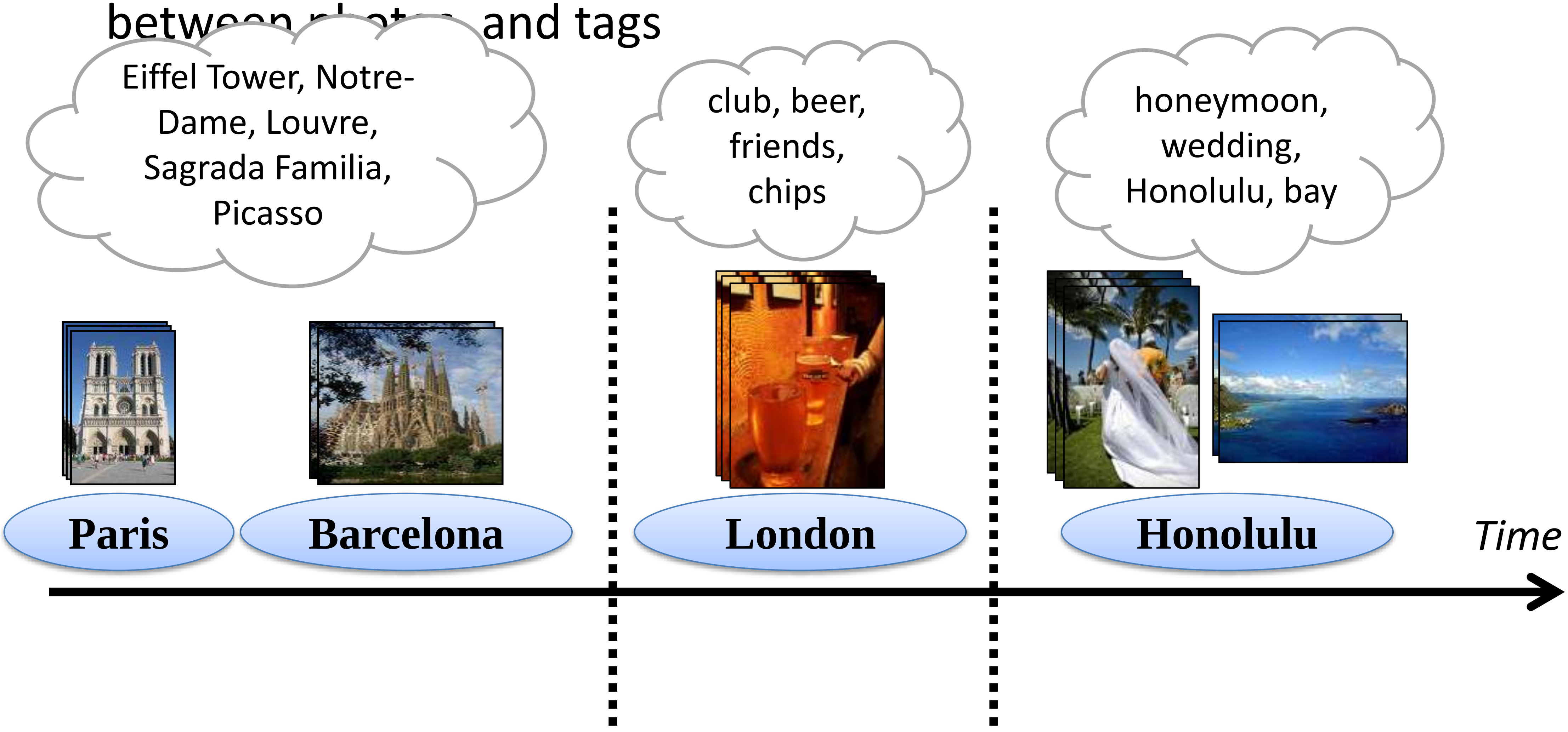
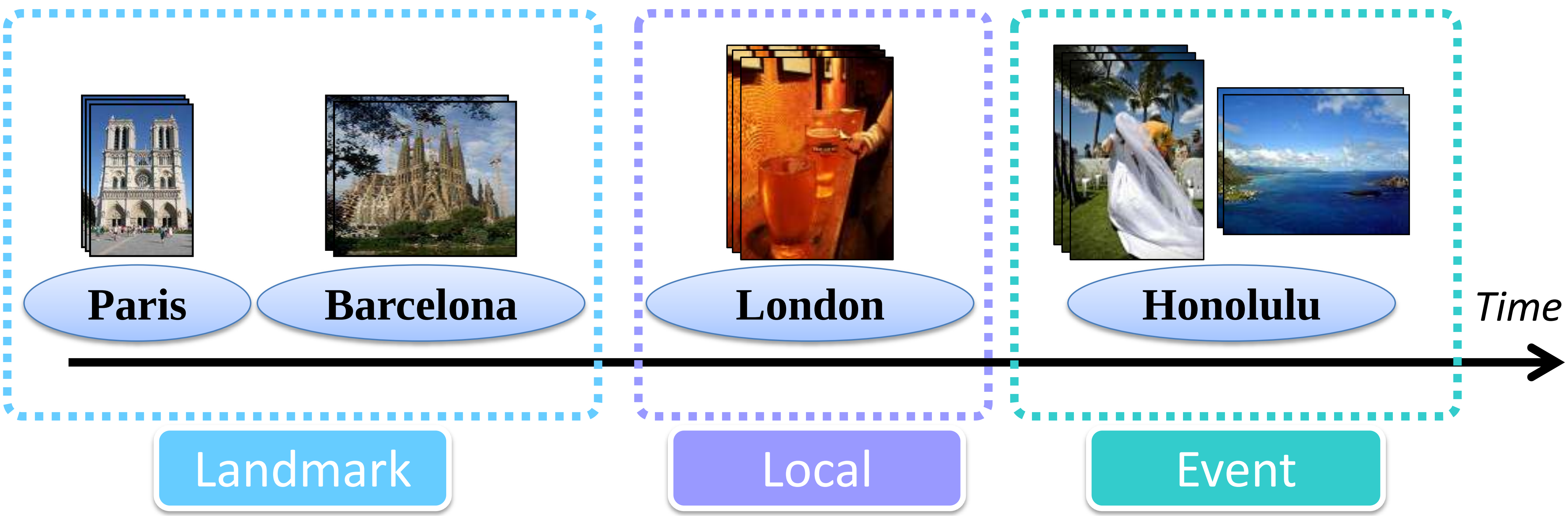


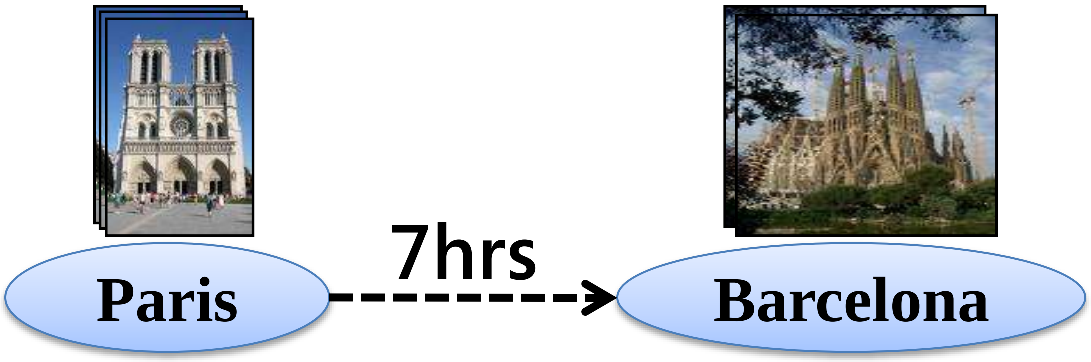
Photo Trip Pattern Mining: Classification

- Classify photo trips into categories by SVMs
  - Landmark/Nature/Gourmet/Event/Business/Local
  - Features: tags and locations



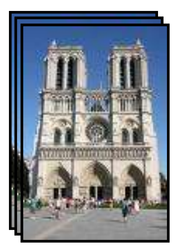
Trip Pattern Mining for Trip Classes

- Apply TAS (Temporary Annotated Sequence) mining algorithm
  - Input: Set of trips extracted from all users
  - Output: Frequent trip patterns, e.g., a set of visited cities and typical transition times.



Trip Semantic Identification

Sky **Louvre Notre-Dame** Paris **Picasso**
London- Eye Summer **Duomo** 2007 Concert
Sforza Castle Wedding



Paris 7hrs **Barcelona**

Trip semantics

Trip Semantic Identification

- Detect descriptive tags for each trip pattern
- TF/IDF based method
  - Tag frequency, inverse tag frequency
  - User frequency
- Consider geographical scale of tags to exclude locally/globally common tags
  - “shop”: globally common tags
  - “Beijing,” “BJ”: locally common tags

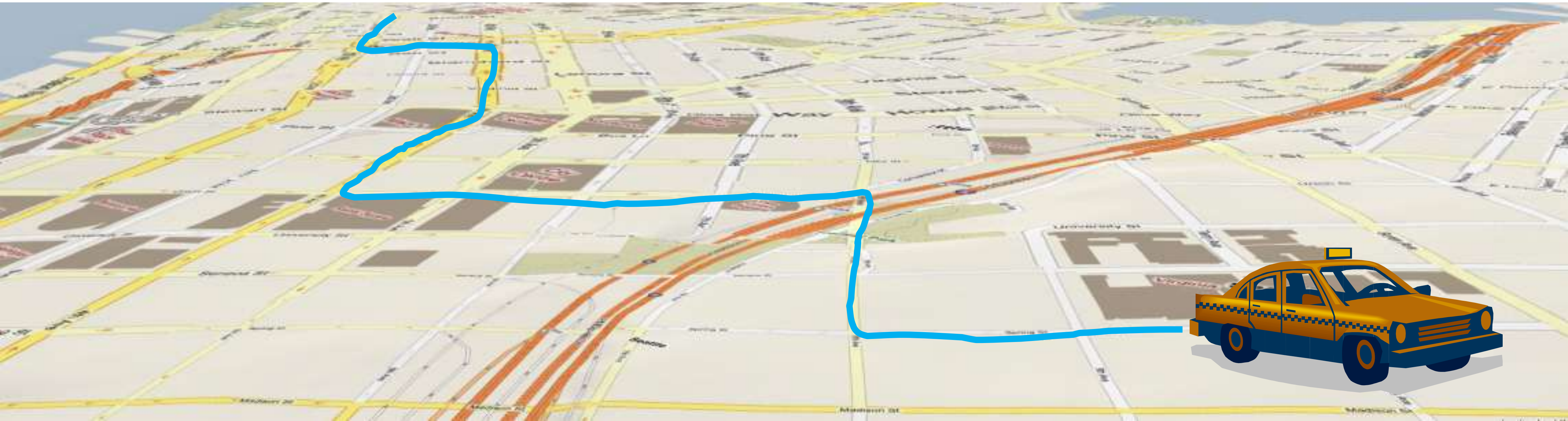
Evaluation

- Collected 5.7 million geo-tagged photos and conducted evaluation
- 72% precision and 85% recall for segmentation detection
- 79% accuracy for trip classification
  - Tags are most dominant feature
  - Combination of tags and locations performed best
  - Locations can compensate photos without tags

Examples

Trip class	Trip pattern	Trip semantics
Landmark	{Paradise, Las Vegas}	casinos, VMA, Bellagio, The Strip, WYNN
Nature	{Sydney, Randwick}	blue sky, barbed wire, inner, bay, Manly
Gourmet	{Camberwell, Melbourne}	cookie, spoon, rice, Colonial hotel, DJ
Event	{Washington D.C, Arlington}	mountain biking, WW, Wednesdays at Wakefield, mountain bike race, racing
Business	{Jersey City, New York, Jersey City}	comedians, MSN, live.com, Steve Kelley, Yahoo
Local	{Boston, Cambridge}	ants, mall, hospital, highway, living room

T-Drive: Driving Directions Based on Taxi Traces



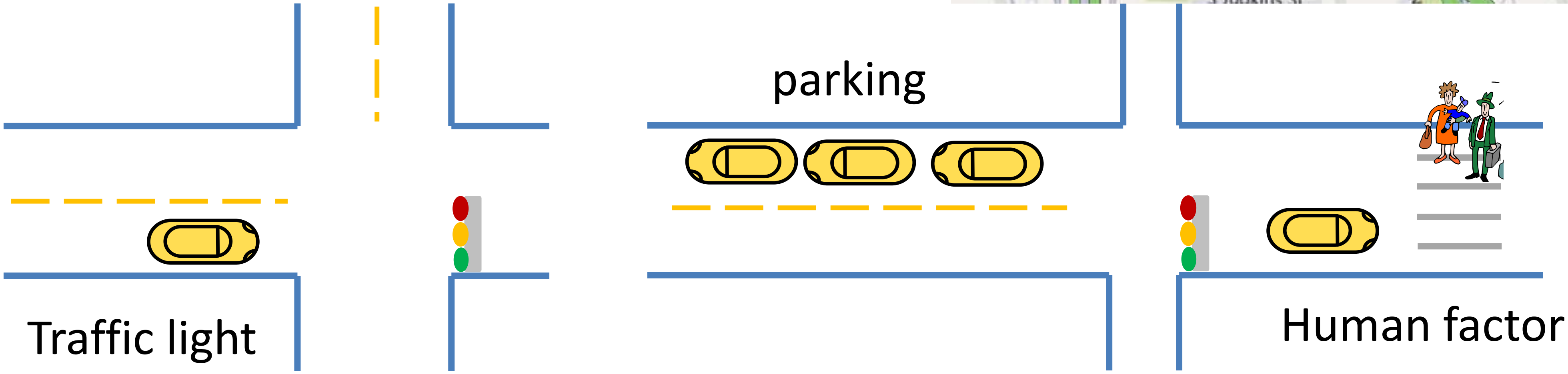
Q=(q_s, q_d and t)

t =7:00am

t = 8:30am

Background

- Shortest path and Fastest path (speed constraints)
- Real-time traffic analysis
  - Methods
    - Road sensors
    - Visual-based (camera)
    - Floating car data
  - Open challenges: coverage, accuracy,…
  - Have not been integrated into routing



Background

- What a drive really needs?

Traffic Estimation (Speed)

Many open challenges

Error Propagation

Sensor Data

Driving Directions

- Finding driving direction > > Traffic analysis

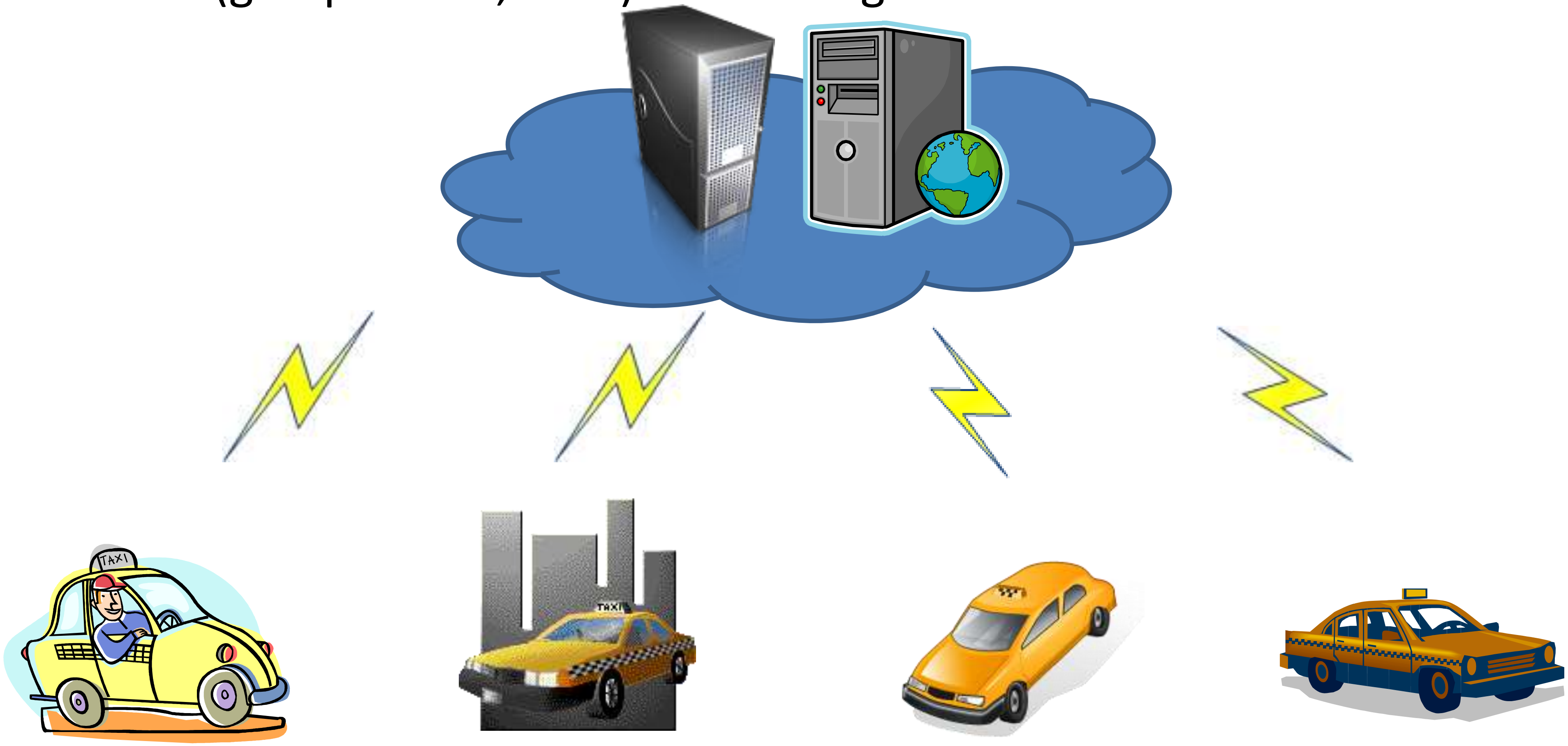
Physical Routes

Traffic flows

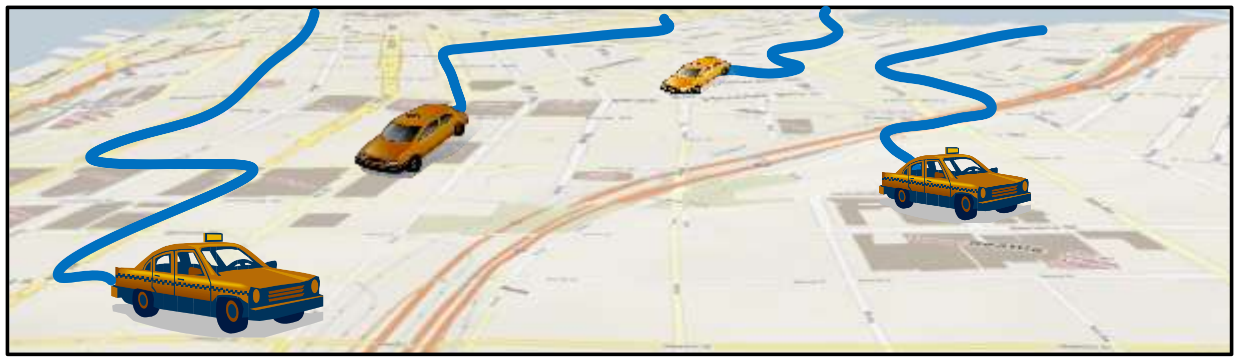
Drivers

Observations

- A big city with traffic problem usually has many taxis
  - Beijing has 70,000+ taxis with a GPS sensor
  - Send (geo-position, time) to a management center



Motivation

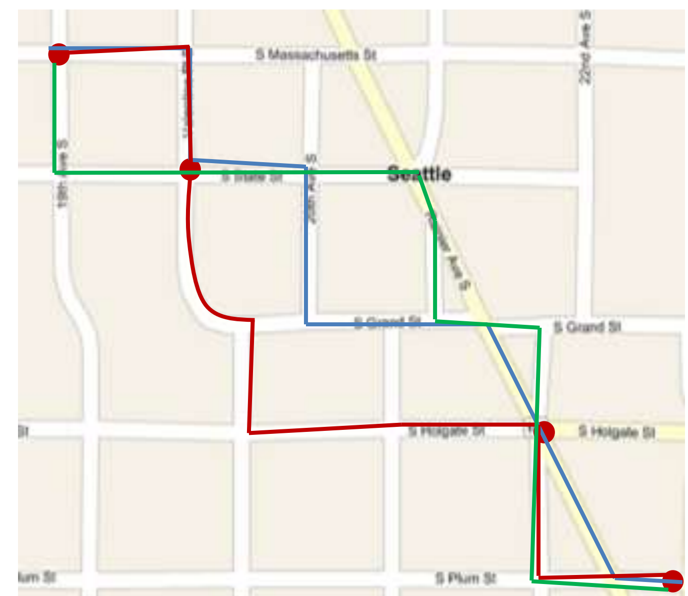


Human Intelligence

Traffic patterns

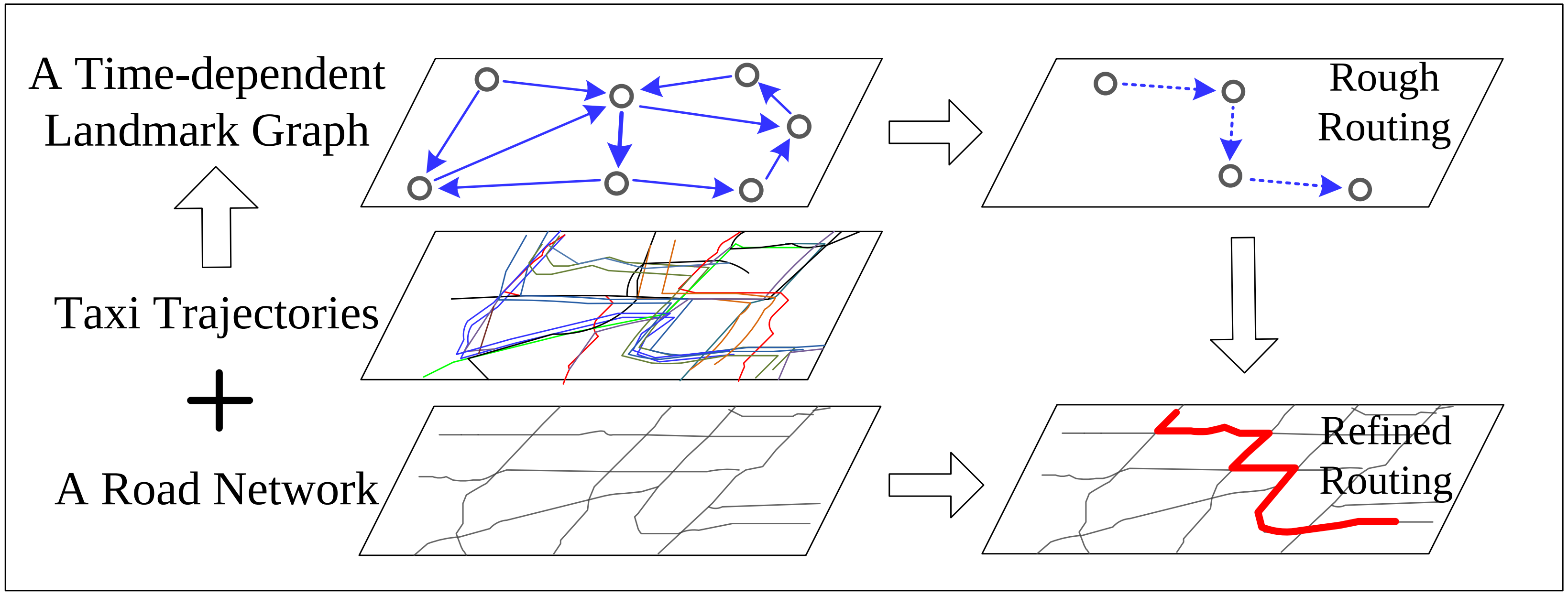
Challenges we are faced

- Intelligence modeling
- Data sparseness
- Low-sampling-rate



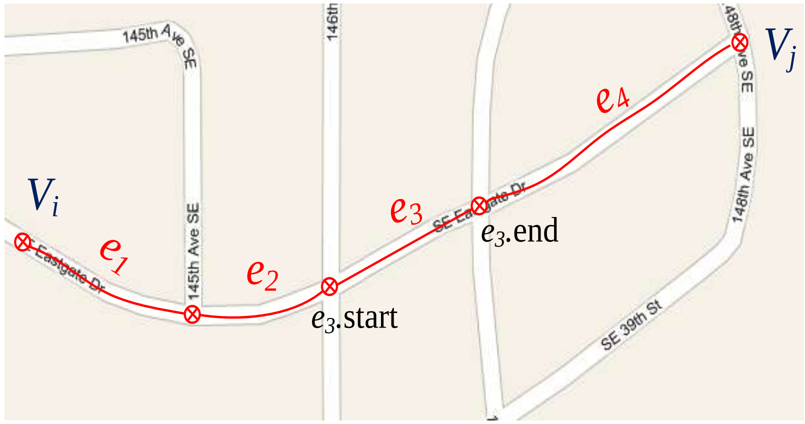
Methodology

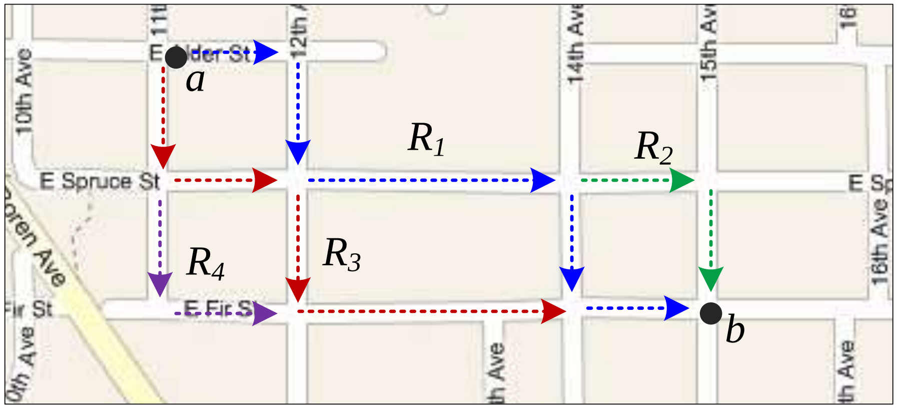
- Pre-processing
- Building landmark graph
- Estimate travel time
- Time-dependent two-stage routing



Step 1: Pre-processing

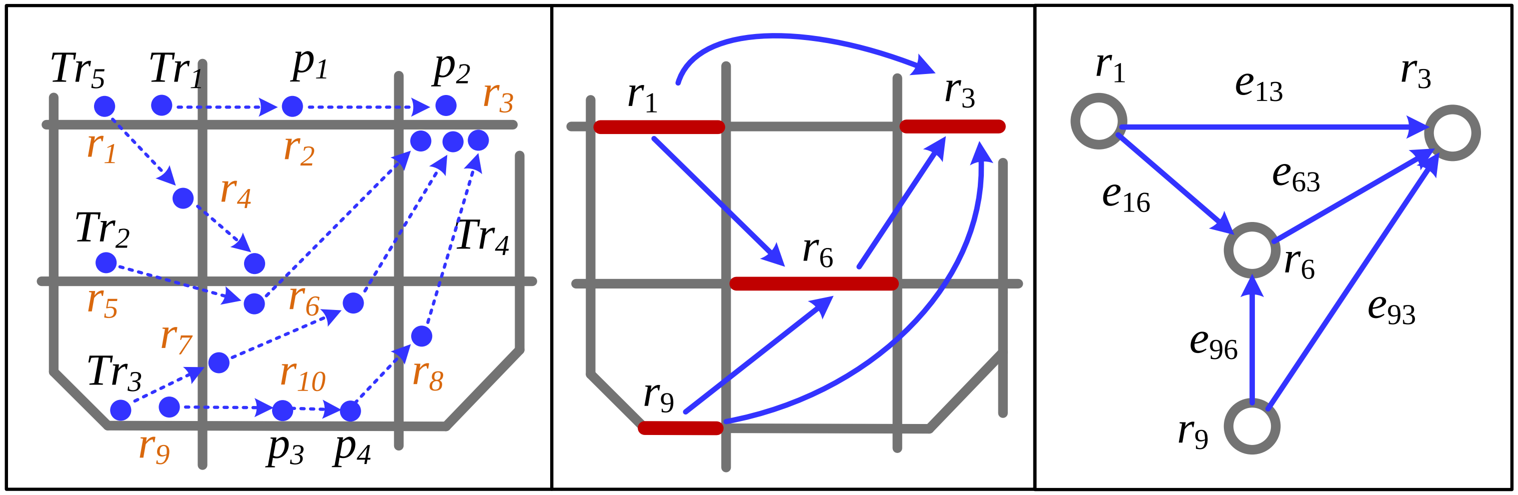
- Trajectory segmentation
  - Find out effective trips with passengers inside a taxi
  - A tag generated by a taxi meter
- Map-matching
  - map a GPS point to a road segment
  - IVMM method (accuracy 0.8, <3min)





Step 2: Building landmark graphs

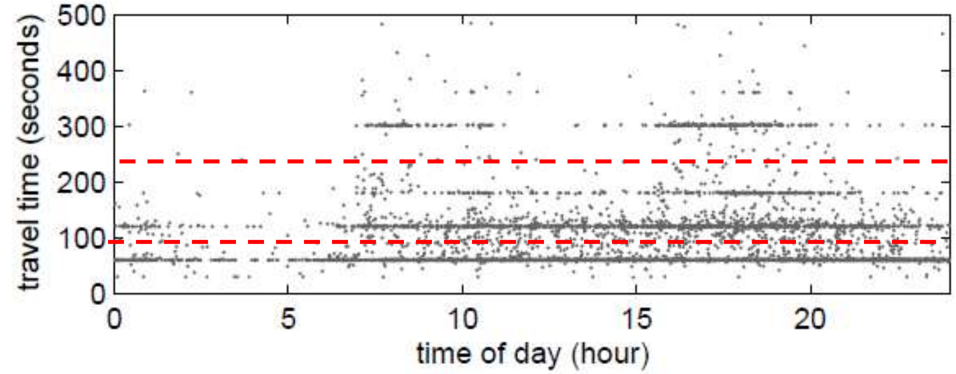
- Detecting landmarks
  - A landmark is a frequently-traversed road segment
  - Top k road segments, e.g. k=4
- Establishing landmark edges
  - Number of transitions between two landmark edges > δ
  - E.g., $\delta = 1$



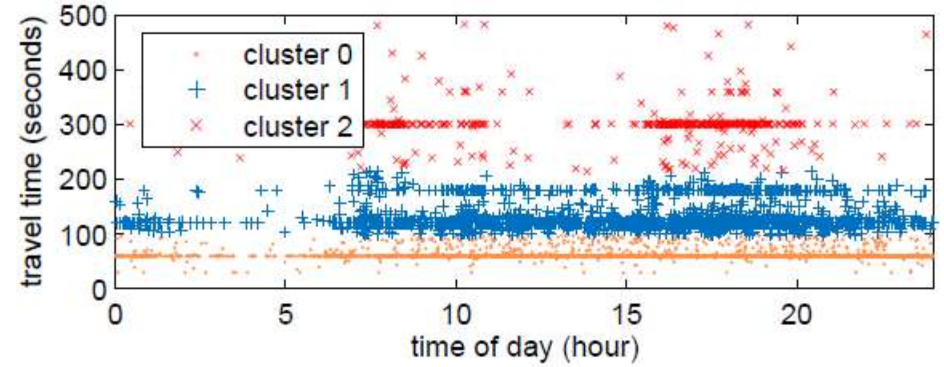
A) Matched taxi trajectories B) Detected landmarks C) A landmark graph

Step 3: Travel time estimation

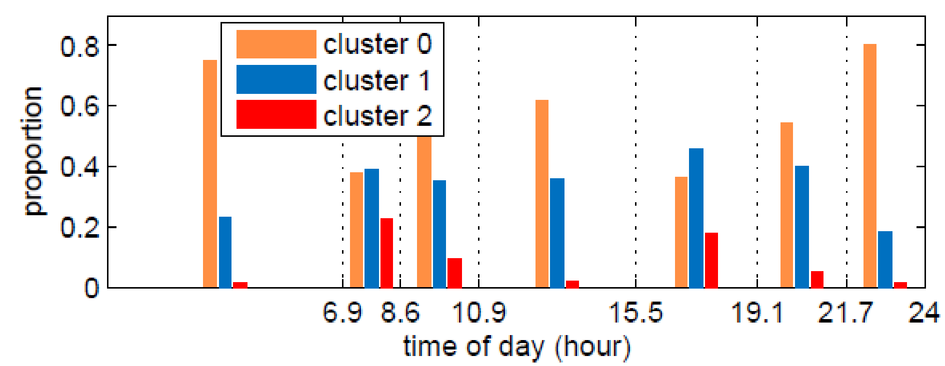
- The travel time of an landmark edge
  - Varies in time of day
  - is not a Gaussian distribution
  - Looks like a set of clusters
- A time-based single valued function is not a good choice
  - Data sparseness
  - Loss information related to drivers
  - Different landmark edges have different time-variant patterns
  - Cannot use a predefined time splits
- VE-Clustering
  - Clustering samples according to variance
  - Split the time line in terms of entropy



(a) Transitions of a landmark Edge

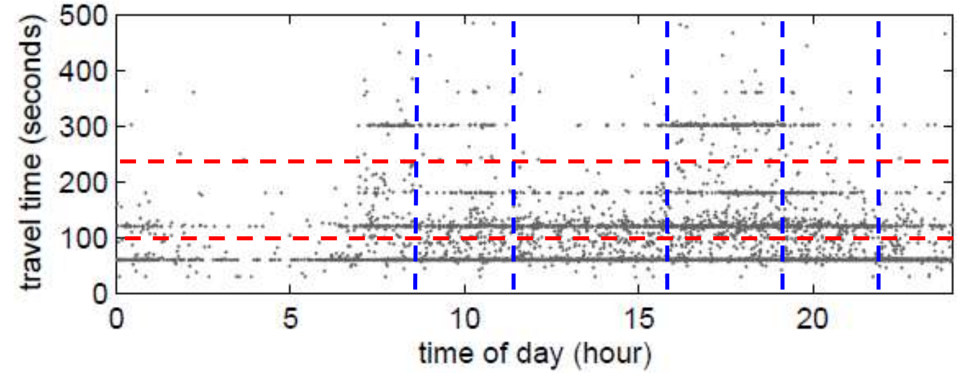


(b) V-Clustering result



(c) VE-Clustering result

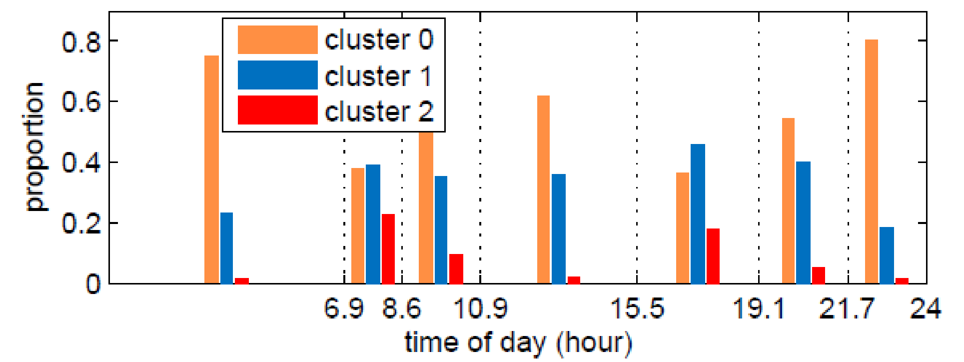
Step 3: Travel time estimation



- V-Clustering
  - Sort the transitions by their travel times
  - Find the best split points on Y axis in a binary-recursive way

$$\mathrm{WAV}(i;L) = \frac{|L_1(i)|}{|L|}\mathrm{Var}(L_1(i)) + \frac{|L_2(i)|}{|L|}\mathrm{Var}(L_2(i))$$

$$\Delta V(i) = \mathrm{Var}(L) - \mathrm{WAV}(i;L).$$



(c) VE-Clustering result

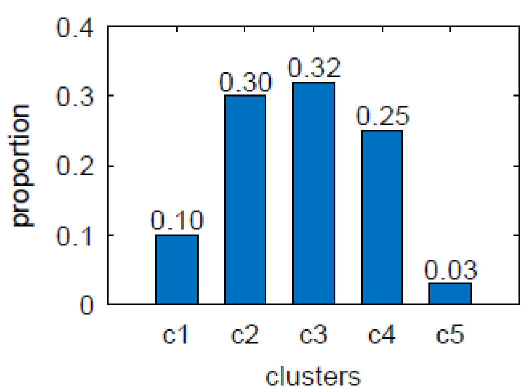
- E-clustering
  - Represent a transition with a cluster ID
  - Find the best split points on X axis

$$\mathrm{WAE}(i;S^{xc}) = \frac{|S_1^{xc}(i)|}{|S^{xc}|}\mathrm{Ent}(S_1^{xc}(i)) + \frac{|S_2^{xc}(i)|}{|S^{xc}|}\mathrm{Ent}(S_2^{xc}(i))$$

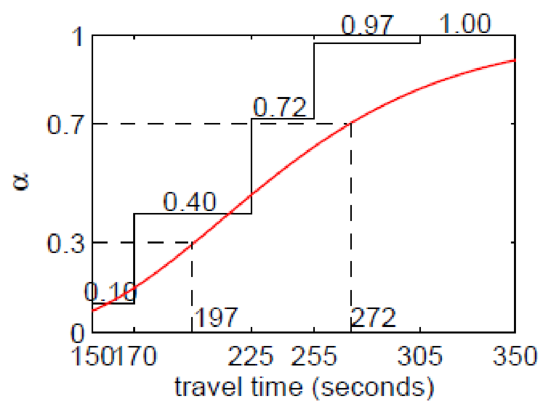
$$\Delta E(i) = \mathrm{Ent}(S^{xc}) - \mathrm{WAE}(i;S^{xc}). \qquad \mathrm{Ent}(S^{xc}) = -\sum_{i=1}^{m} p_i \log(p_i)$$

Step 4: Two-stage routing

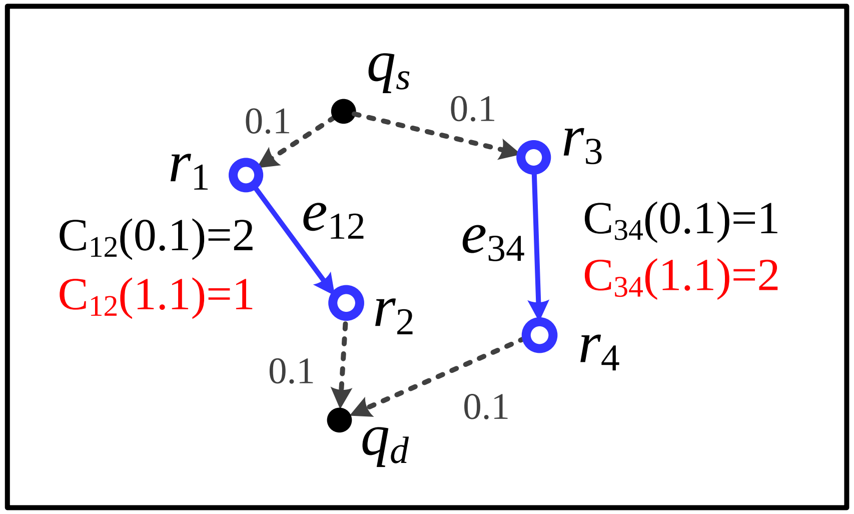
- Rough routing
  - Search a landmark graph for
  - A rough route: a sequence of landmarks
  - Based on a user query (q_s, q_d, t, α)
  - Using a time-dependent routing algorithm



(a) Travel time distribution

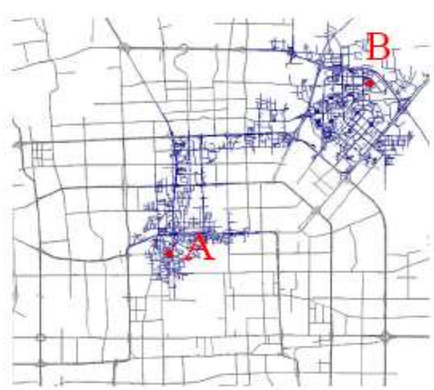


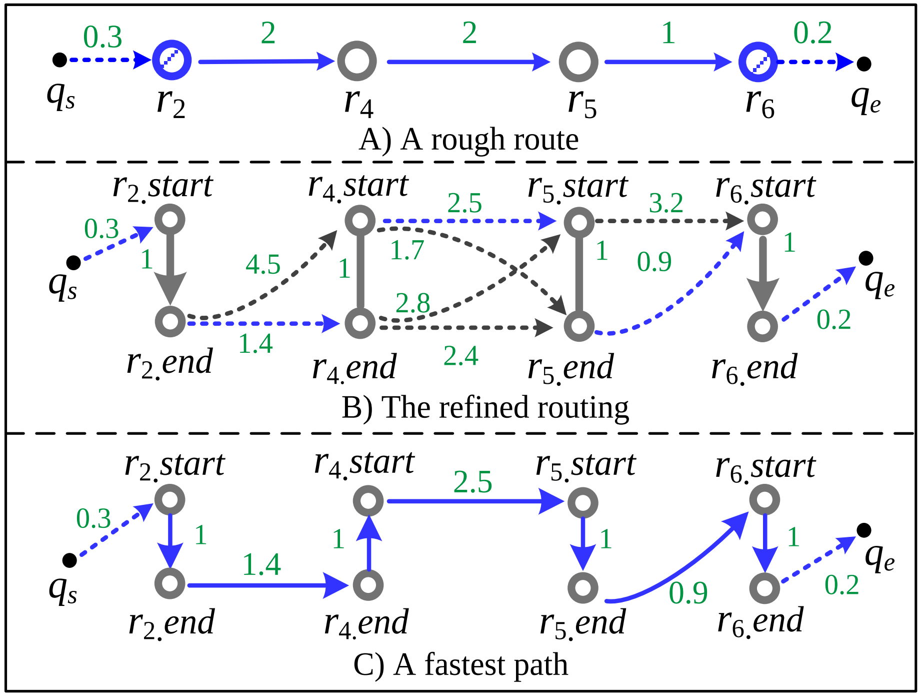
(b) Cumulative frequency



Step 4: Two-stage routing

- Refined routing
  - Find out the fastest path connecting the consecutive landmarks
  - Can use speed constraints
  - Dynamic programming
- Very efficient
  - Smaller search spaces
  - Computed in parallel



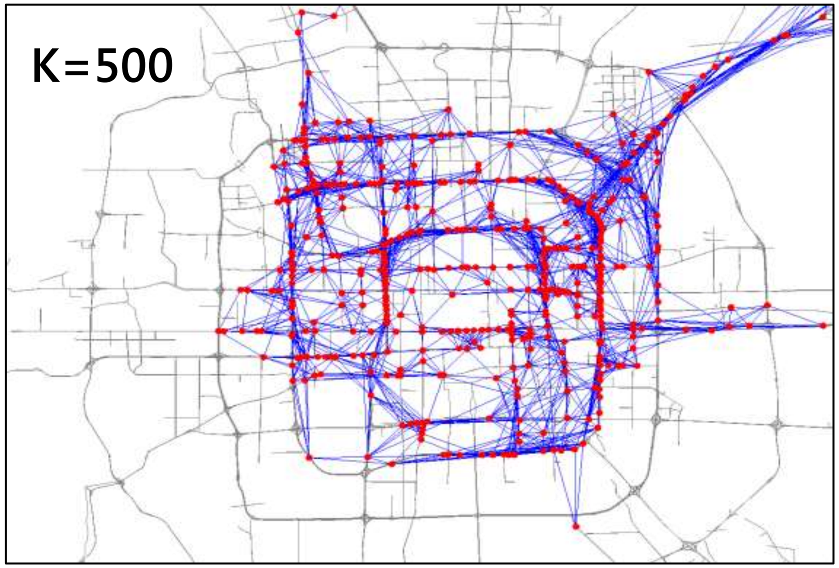


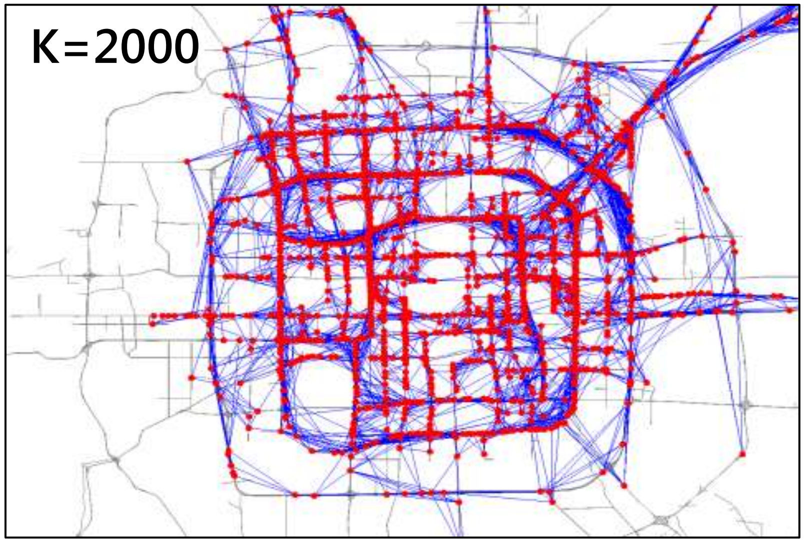
Implementation & Evaluation

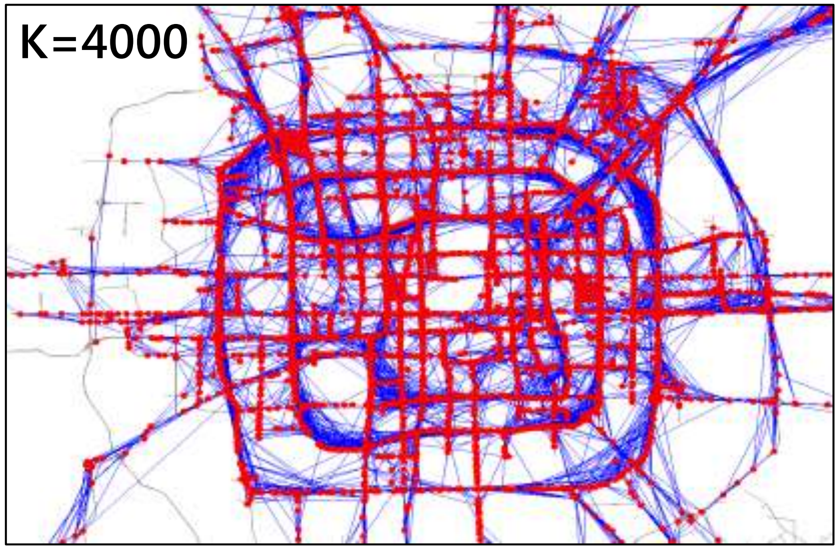
- **6-month** real dataset of **30,000 taxis** in Beijing
  - Total distance: almost 0.5 billion (446 million) KM
  - Number of GPS points: almost 1 billion (855 million)
  - Average time interval between two points is 2 minutes
  - Average distance between two GPS points is 600 meters
- Evaluating landmark graphs
- Evaluating the suggested routes by
  - Using synthetic queries
  - In the field studies

Evaluating landmark graphs

- Estimate travel time with a landmark graph
- Using real-user trajectories
  - 30 users' driving paths in 2 months
  - GeoLife GPS trajectories (released)

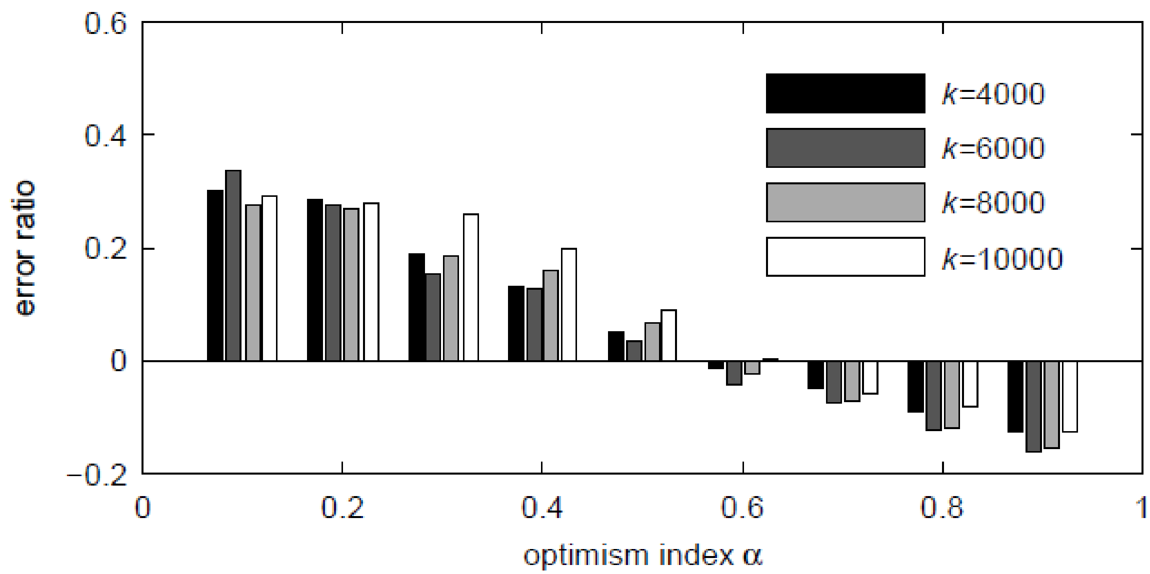


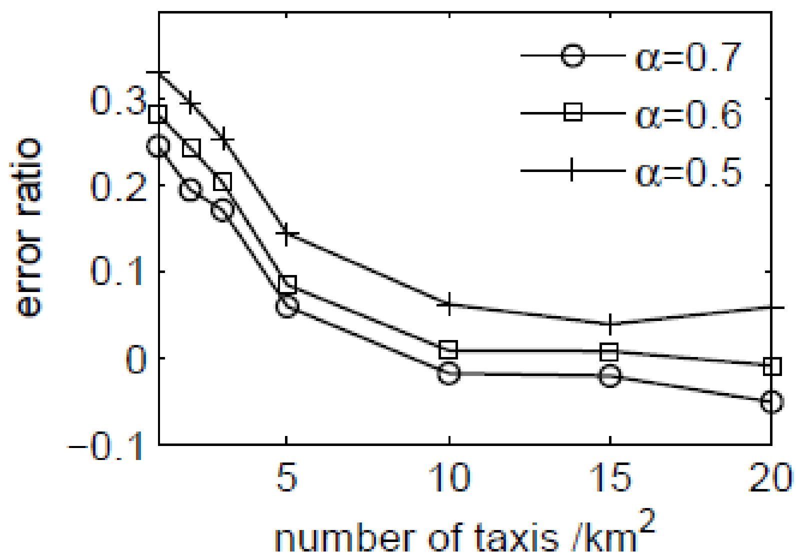




Evaluating landmark graphs

- Accurately estimate the travel time of a route
- 10 taxis/ km^2 is enough





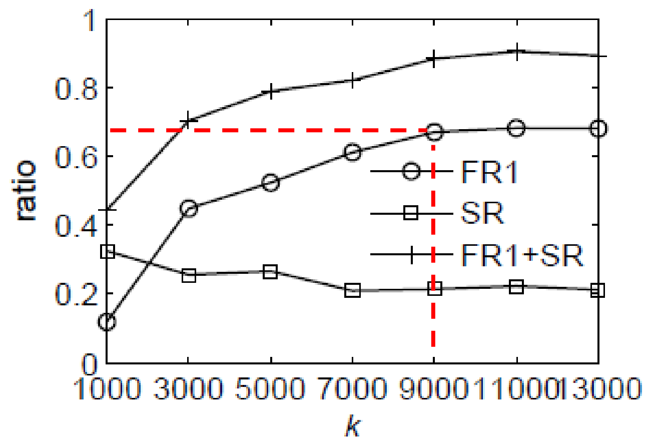
(b) k=10000

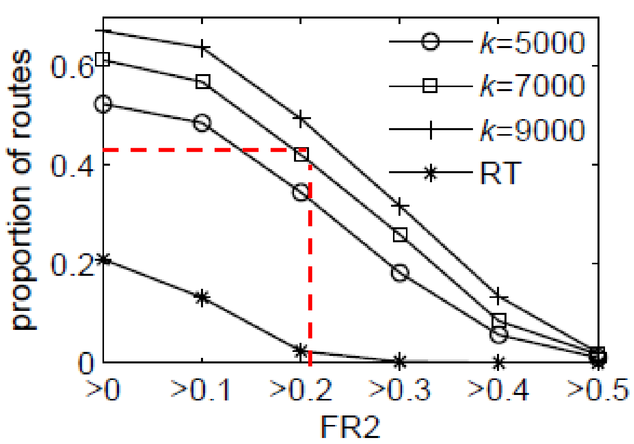
Synthetic queries

- Baselines
  - Speed-constraints-based method (SC)
  - Real-time traffic-based method (RT)
- Measurements
  - FR1, FR2 and SR
  - Using SC method as a basis

$$\mathrm{FR1} = \frac{\text{Number(A's travel time<B's travel time)}}{\text{Number(queries)}}$$

$$\mathrm{FR2} = \frac{\text{B's travel time}-\text{A's travel time}}{\text{B's travel time}}.$$





	α	k	FR1	SR
TDrive	0.4	6,000	0.509	0.281
	0.4	9,000	0.647	0.222
	0.6	6,000	0.511	0.272
	0.6	9,000	0.653	0.216
	0.7	6,000	0.544	0.227
	0.7	9,000	0.672	0.214
RT approach			0.206	0.671

In the field study

- Evaluation 1
  - Same drivers traverse
  - different routes at different times
- Evaluation 2
  - Different two users with similar driving skills
  - Travers two routes simultaneously

Table 1: Trajectories of the In-the-field Study

	Evaluation 1	Evaluation 2
Num. Trajectories	360	60
Num. Users	30	2
Total Distance (km)	5304	814
Total Duration (hour)	165.24	25.09
Evaluation Days	10	6

Table 5: In-the-field Evaluation 1

	T-Drive	Google	△	R1	R2
Distance	13.91km	15.56km	1.65km	0.517	0.106
Duration	25.80min	29.28min	3.48min	0.808	0.119

Table 6: In-the-field Evaluation 2

	T-Drive	Google	△	R1	R2
Distance	13.58km	13.55km	-0.03km	0.367	-0.002
Duration	23.18min	27.00min	3.82min	0.750	0.141
WaitTime	4.77min	6.50min	1.73min	0.633	0.267

Results

- **More effective**
  - **60-70%** of the routes suggested by our method are faster than Bing and Google Maps.
  - Over **50%** of the routes are **20+%** faster than Bing and Google.
  - On average, we save **5** minutes per 30 minutes driving trip.
- More efficient
- More functional

Conclusions

- Build intelligence from the physical world
  - Activity/location recommendation based on GPS trajectories
  - Mining geo-tagged photos for travel recommendation
  - Driving directions based on taxi traces

- Challenges and future directions
  - How to protect privacy?
  - How to support real-time information sharing and search?
  - How to reduce energy consumption?

UbiComp 2011 in Beijing: weibo.com/ubicomp2011

Date: Sep. 17-21, 2011 **Venue: Tsinghua University**

Chairs: **Yuanchun Shi (Tsinghua), James Landay (UW/MSR)**

Program Chairs: **Don Patterson (UCI), Yvonne Rogers (OU), Xing Xie (MSR)**

Thanks!

Xing Xie
Microsoft Research Asia
Aug. 30, 2011