Machine Learning for Spatial Data Management and Location-Based Services

Hua Lu (Professor, PhD)
Aalborg University (Copenhagen), Denmark
luhua@cs.aau.dk https://www.cs.aau.dk/~luhua

Agenda

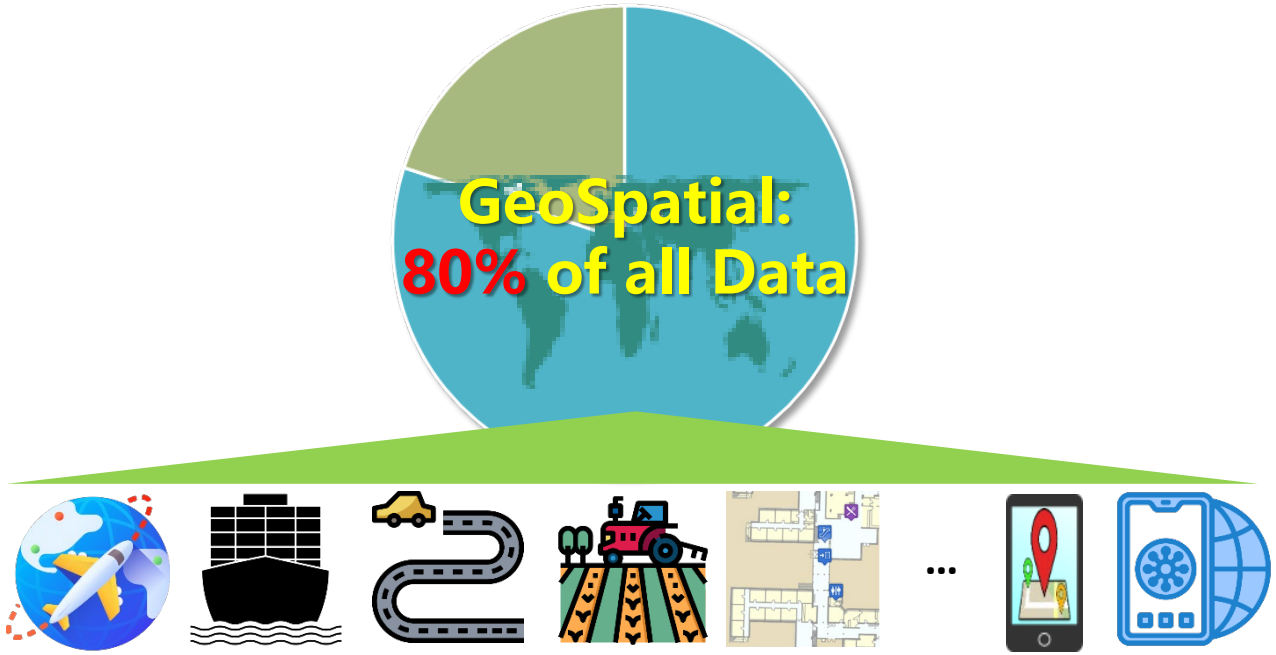
- Introduction
- Learned index for spatial data
- Representation learning of spatial data
- Machine learning for location-based services
- Conclusion and future research

Changes/adjustments from advertised

- Representation learning added
- Location inference in SoMe omitted

Pervasive Spatial Data

- Spatial data management
  - Efficient value creation from spatial data
  - Foundation for location-based services (LBS)



Created by ChatGPT

Challenges of Big Spatial Data

- Volume: The *amounts* of spatial data are continuously increasing. (80%)
- Veracity: Spatial data is often *dirty*, with missing values, redundancies and wrong values. (uncertain positioning)
- Velocity: In many cases, spatial data updates often. (positioning and monitoring)
- Variety: Spatial data is of different types, but it is desirable if they can be handled in the same manner. (raster vs. vector data; points, polylines, polygons, and trajectories)

- How to create value in the presence of these challenges?
  - Traditional techniques often fall short. We turn to machine learning (ML).

Topics of this lecture

- Learned index for spatial data
  - Volume and Velocity
- Representation learning of spatial data
  - Veracity and Variety
- Machine learning for location-based services
  - Veracity and Velocity

Agenda

- Introduction
- Learned index for spatial data
  - Preliminaries
  - ZM-index
  - LISA
- Representation learning of spatial data
- Machine learning for location-based services
- Future research directions

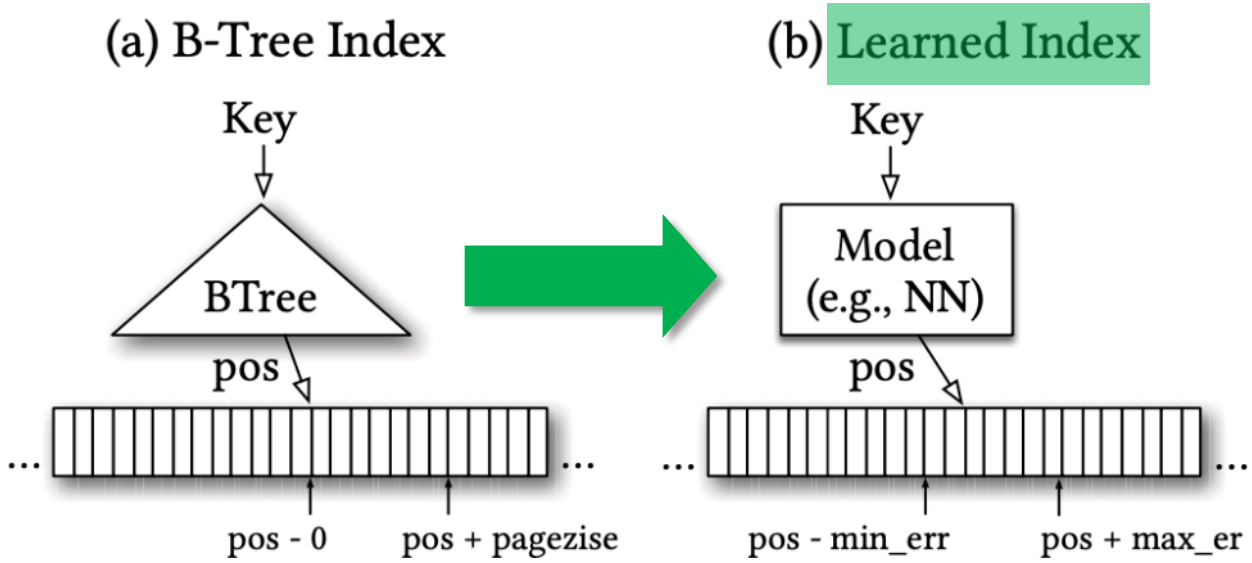
Agenda

- Introduction
- Learned index for spatial data
  - Preliminaries
  - ZM-index
  - LISA
- Representation learning of spatial data
- Machine learning for location-based services
- Conclusion and future research

B-Trees Viewed as Regression Models

Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, Neoklis Polyzotis: The Case for Learned Index Structures. ACM SIGMOD 2018.

- Assumptions:
  - Static data with 1D keys *sorted* in an in-memory array
  - Tree leaf nodes are pieces in the array.
- Given a search key, B-tree *predicts* a position for it
  - Error bound: [0, page_size]
- B-tree can be replaced by an ML model (**Learned Index**)
  - The key is guaranteed to be found within the error bound if it does exit.

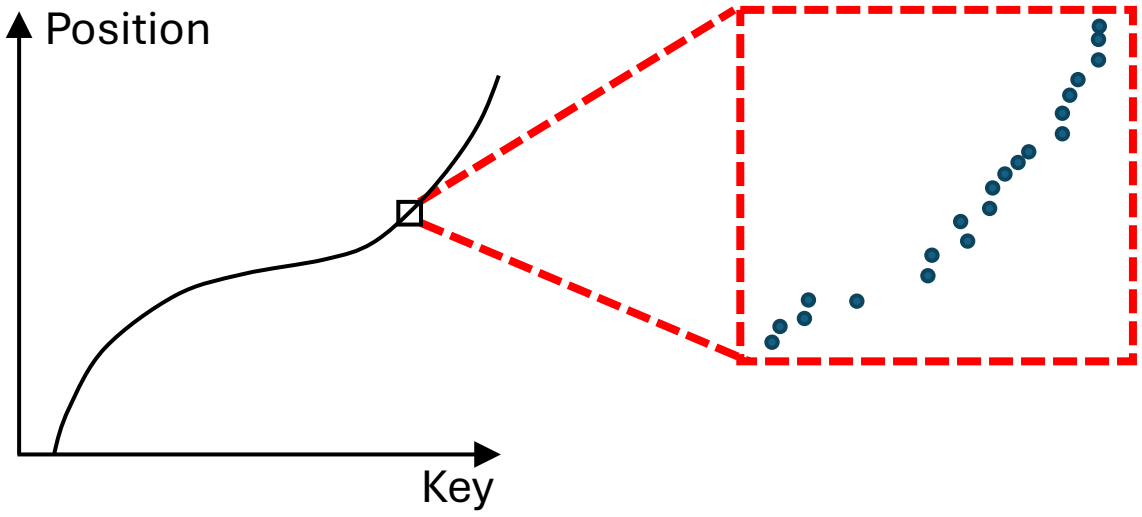


B-Tree as A CDF Model

- Model Input: A search key
- Model Output: The key's position in the sorted data array
- **Observation**: A model that predicts a key's position in a sorted array *approximates* the cumulative distribution function (CDF).

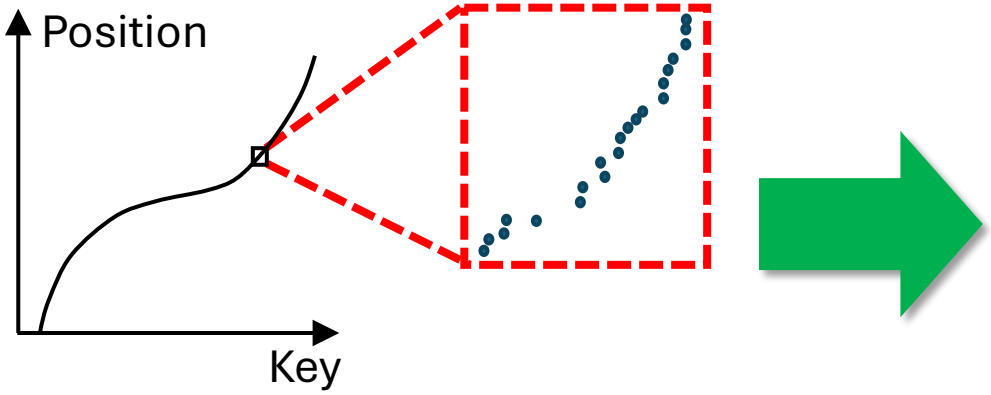
  $p = F(\text{Key}) * N$

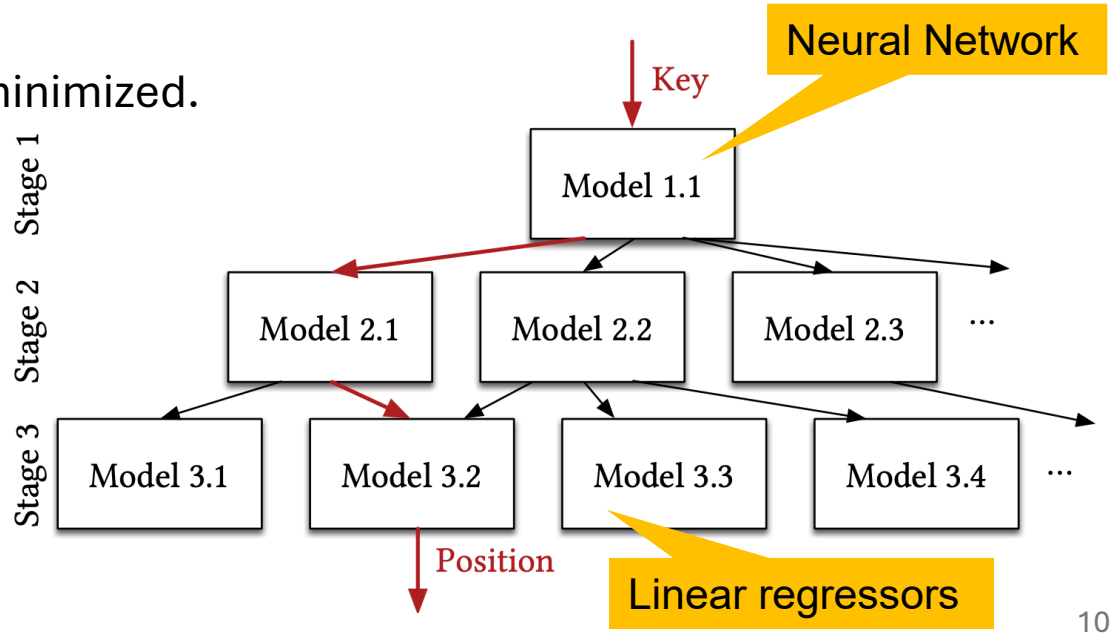
  - p: Predicted position
  - F: CDF $P(X \leq \text{Key})$
  - N: Total number of keys
- Local irregularities in CDF
  - A single model does not fly



Recursive Model Index (RMI): A Learned Index

- A hierarchy of models
  - A non-leaf node model predicts which child model to use for a key.
  - A leaf node model predicts the key's position in the *sorted* in-memory array.
    - with an error bound
  - Each model is only responsible for a range of keys.
    - Lower models handle narrower ranges
    - The effect of local irregularities is thus minimized.





Advantages of Learned Index

- Learned index (model)
  - Less storage (model parameters)
  - Distribution-aware
  - Fast(er) queries
  - Efficient updates *
- Traditional index
  - Large storage
  - Distribution-independent
  - Slow queries
  - Inefficient updates

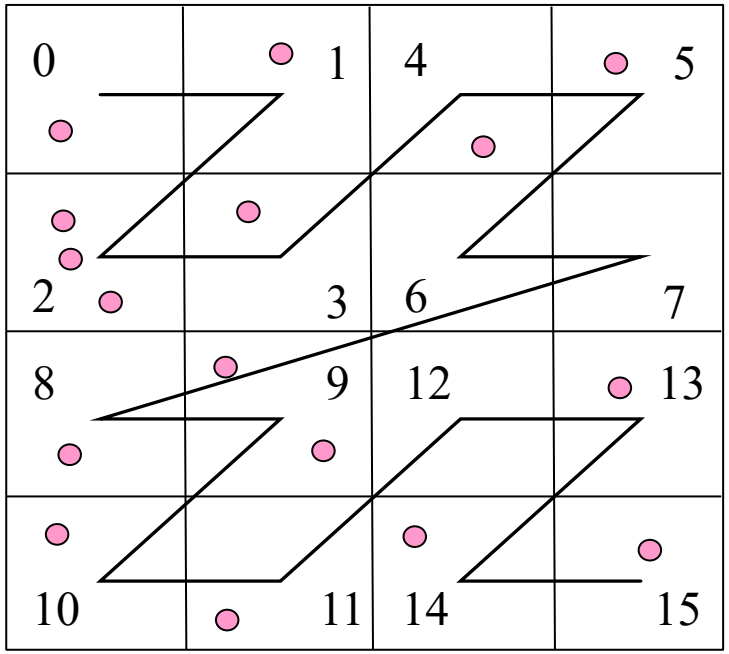
- What if keys are not 1D but multi-D spatial points?
  - Z-Order Model Index (ZM)
  - LISA

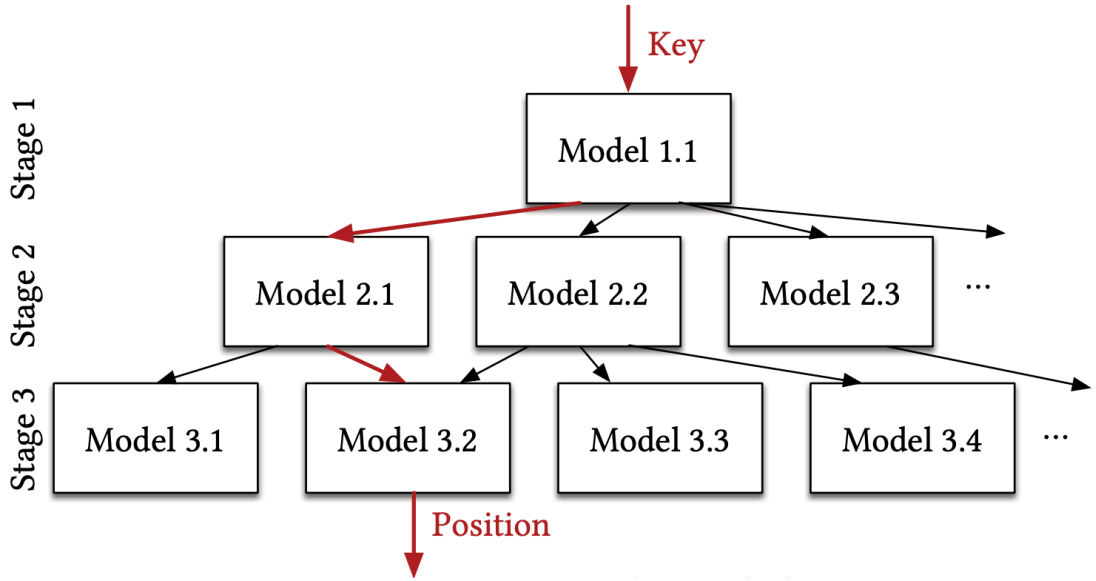
Agenda

- Introduction
- Learned index for spatial data
  - Preliminaries
  - ZM-index
    (*Joint work with HKBU's DB Group*)
  - LISA
- Representation learning of spatial data
- Machine learning for location-based services
- Conclusion and future research

Idea behind Z-Order Model Index (ZM)

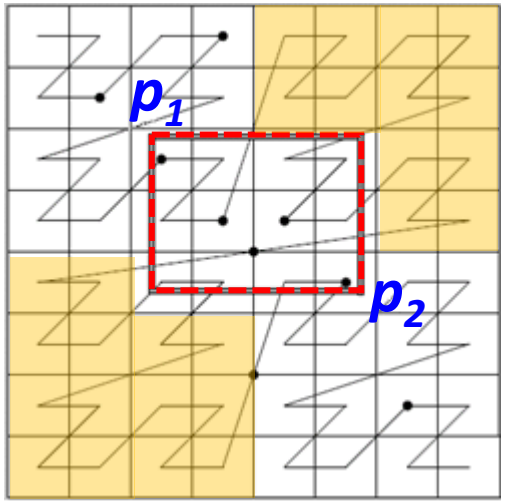
- Each 2D point is transformed into a 1D key by a Z-order curve.
  - Many-to-one, depending on the grid granularity
- All such 1D keys are simply indexed by RMI.





Range Query via ZM

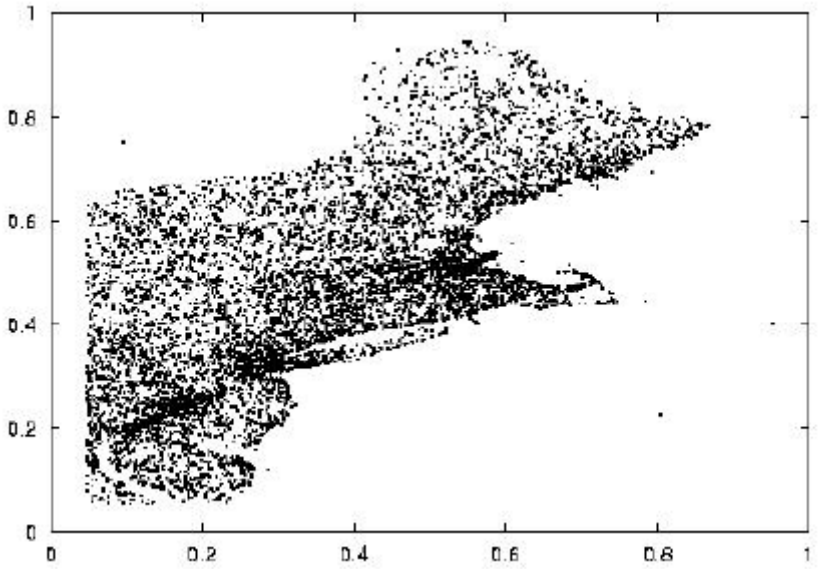
- A range query on the original points
  - A range is represented by two corner points p_1 and p_2.
- The 2D range is transformed to a range of 1D keys.
  - In this example, [12, 49]
- All points of such 1D keys are found by a *single* range query via the RMI.
- Refinement is needed in the original space.
  - To check if a point really falls in the query range.
- In this example, many cells and points are accessed *unnecessarily*.



	0	1	2	3	4	5	6	7
0	0	1	4	5	16	17	20	21
1	2	3	6	7	18	19	22	23
2	8	9	12	13	24	25	28	29
3	10	11	14	15	26	27	30	31
4	32	33	36	37	48	49	52	53
5	34	35	38	39	50	51	54	55
6	40	41	44	45	56	57	60	61
7	42	43	46	47	58	59	62	63

Performance Evaluation

- Datasets
  - RANDOM: Synthetic dataset with 100,000 randomly generated objects in a square Euclidean space.
  - POST: Real-world dataset of the positions of 123,593 post offices in the northeast of America.
- Experiments
  - Index size
  - Point query
  - Range query



Evaluation Results

- ZM compared to R-tree
  - Significantly smaller index size
  - Much faster point queries
  - *Slightly* faster range queries

Fig. 5: Index memory size

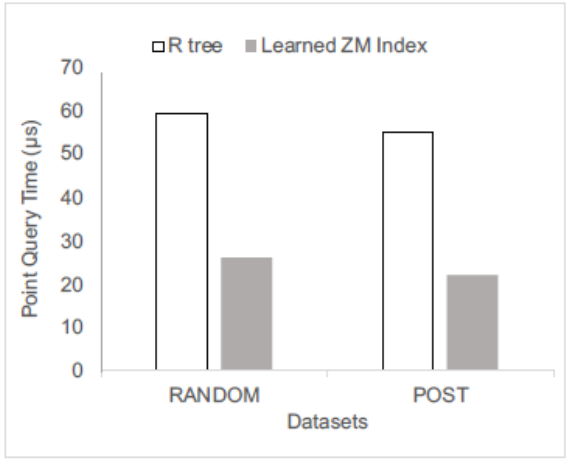
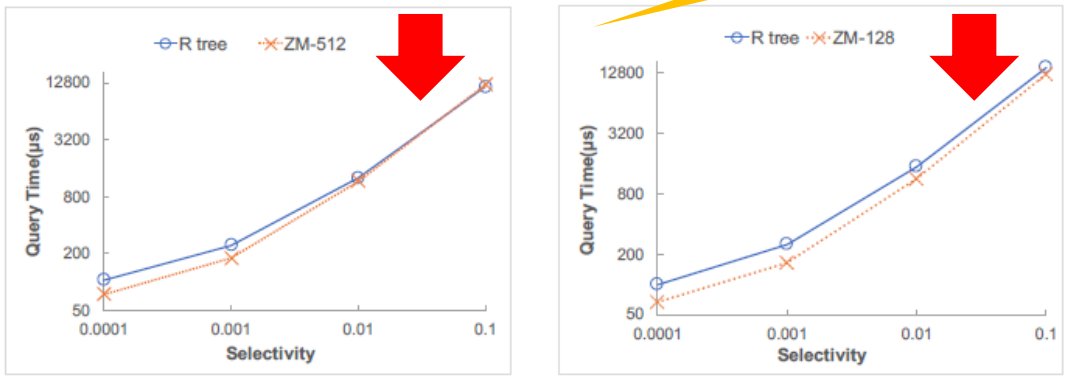


Fig. 6: Point query

Unnecessary cell accesses.



(a) RANDOM (b) POST

Fig. 7: Range Query

Agenda

- Introduction
- Learned index for spatial data
  - Preliminaries
  - ZM-index
  - LISA
    (*Joint work with ZJU*)
- Representation learning of spatial data
- Machine learning for location-based services
- Conclusion and future research

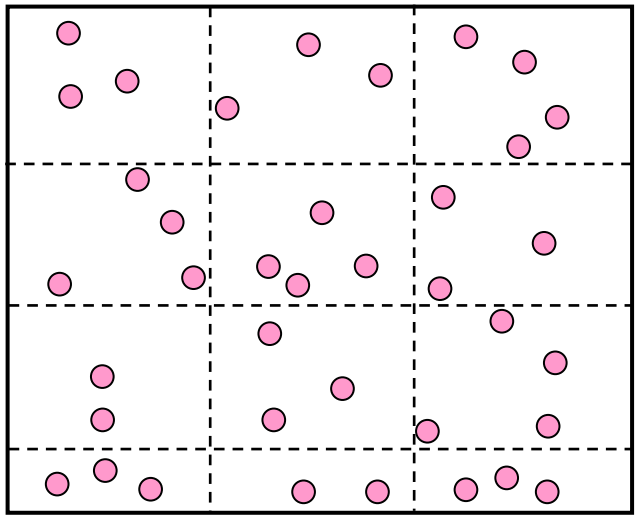
Idea behind LISA

- LISA: A Learned Index Structure for Spatial Data

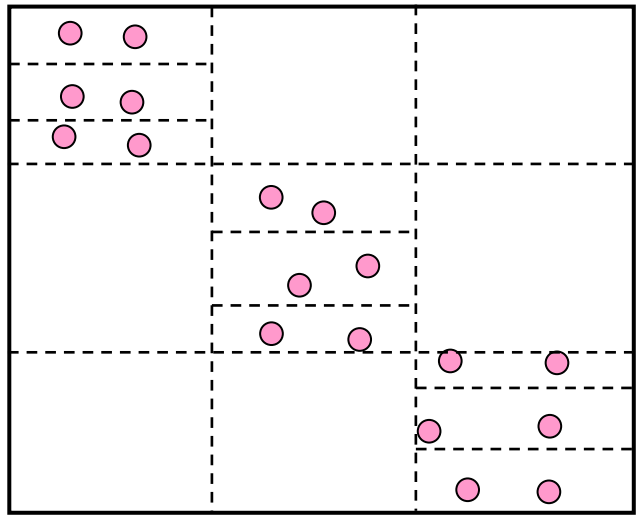
1. Partition the data space into grid **cells**
2. Build a partially monotonic mapping function $\mathcal{M}$ that maps a point (key) to a 1D value
3. Learn a monotonic shard prediction function $\mathcal{SP}$ that assigns to each mapped value a **shard** id
4. For each shard, build a local model to store the keys in **pages**

Step 1: Grid Cells Generation

- Original partitioning strategy
  - In each dimension, partition the space such that the data points are evenly covered by each part
- Modified partitioning strategy
  - After first dimension, find an even partitioning in each part

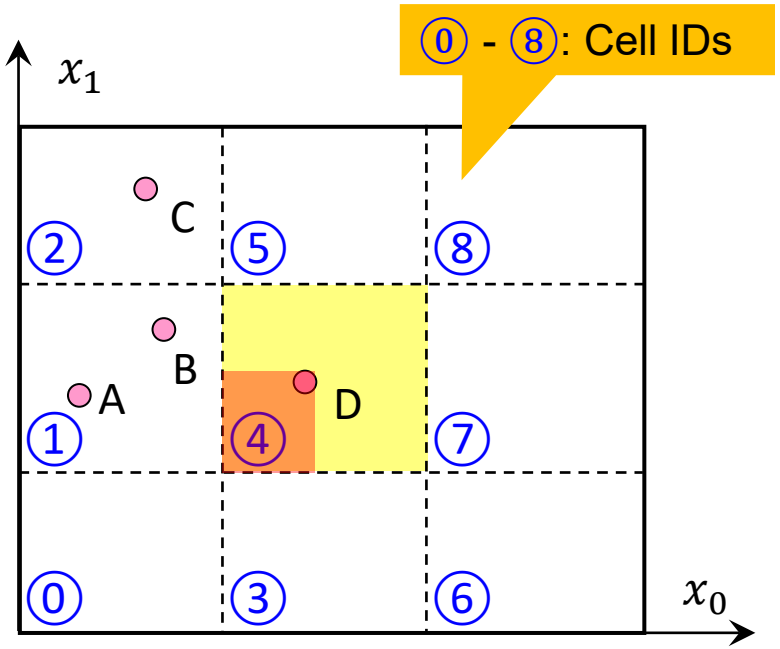


We want that each cell C_i has nearly the same number of points.



Step 2: Mapping Function $\mathcal{M}$

- $\mathcal{M}: \mathbb{R}^d \rightarrow \mathbb{R}$ maps a point (key) to a 1D value with two properties.
- **Property 1**: Partially monotonic
  - $\mathcal{M}(\boldsymbol{x} = (x_0, \ldots, x_{d-1})) \leq \mathcal{M}(\boldsymbol{y} = (y_0, \ldots, y_{d-1}))$ when $x_0 \leq y_0, \ldots, x_{d-1} \leq y_{d-1}$
  - E.g., $\mathcal{M}(\mathrm{A}) \leq \mathcal{M}(\mathrm{B})$
- **Property 2**: Order of grid cells
  - $\mathcal{M}(\boldsymbol{x_i}) \leq \mathcal{M}(\boldsymbol{x_j})$ when $i < j$, where $\boldsymbol{x_i} \in C_i$ and $\boldsymbol{x_j} \in C_j$.
  - E.g., $\mathcal{M}(\mathrm{A}) \leq \mathcal{M}(\mathrm{D}), \mathcal{M}(\mathrm{B}) \leq \mathcal{M}(\mathrm{D}), \mathcal{M}(\mathrm{C}) \leq \mathcal{M}(\mathrm{D})$
- $\mathcal{M}(\boldsymbol{x}) = i + \mu(H_i)/\mu(C_i)$
  - $\boldsymbol{x} = (x_0, \ldots, x_{d-1})$
  - $C_i = [l_0, u_0] \times \cdots [l_{d-1}, u_{d-1}]$
  - $H_i = [l_0, x_0] \times \cdots [l_{d-1}, x_{d-1}]$
  - μ: Lebesgue measure (hypervolume)

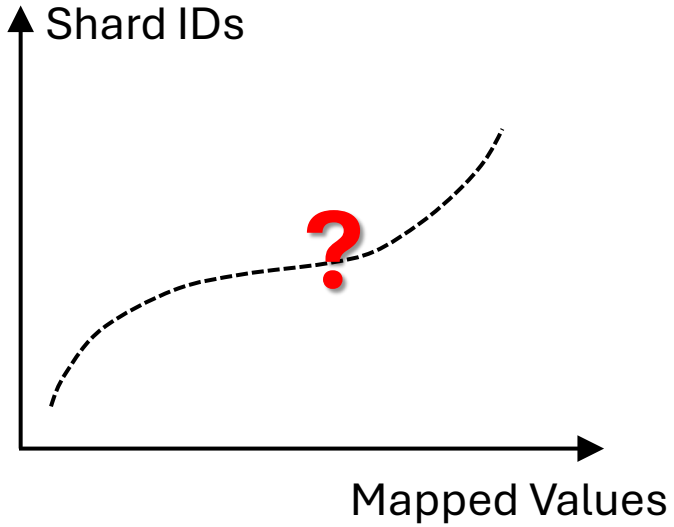


Generated Data Layout in LISA

- In RMI
  - The data is stored in memory.
  - The data layout is already fixed before the regression model is trained.
  - An error means a *local* search that is not so expensive in memory.
- In LISA for spatial points
  - The data is stored in disk.
  - An error in the model can mean loading several extra disk pages (larger IOs).
  - Therefore, based on the mapping function, we design and train a model that directly arranges the data layout in disk pages.

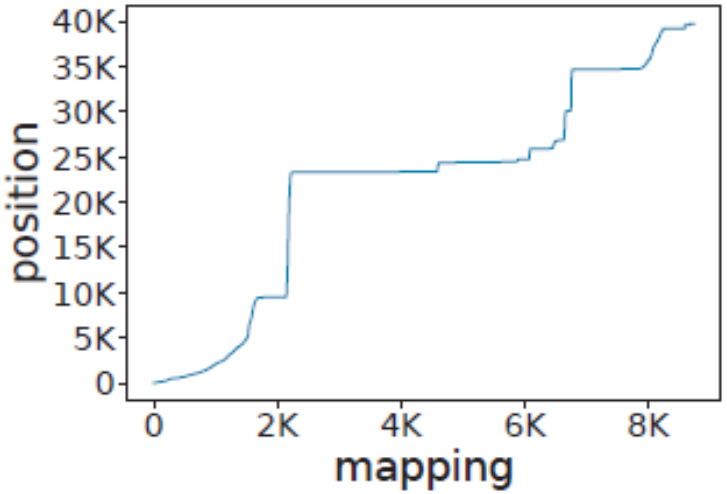
Step 3: Shard Prediction Function $\mathcal{SP}$

- A shard S is a subset of points: The *preimage* of $[a, b) \subseteq [0, +\infty)$ under $\mathcal{M}$
  - $S = \mathcal{M}^{-1}([a, b))$
- Shard prediction function
  - **Purpose**: To determine which points should be assigned with the same shard id. Those points should be stored together in disk space.
  - $\mathcal{SP}: \mathbb{R} \rightarrow [0, +\infty)$
    - **Input**: A mapped value from $\mathcal{M}(\boldsymbol{x})$
    - **Output**: $\lfloor \mathcal{SP}\left(\mathcal{M}(\boldsymbol{x})\right) \rfloor$ is the shard id for $\boldsymbol{x}$
- $\mathcal{SP}$ is a regression model
  - Between mapped values and shard ids
  - Does a single regression function $\mathcal{F}(\boldsymbol{x})$ work?



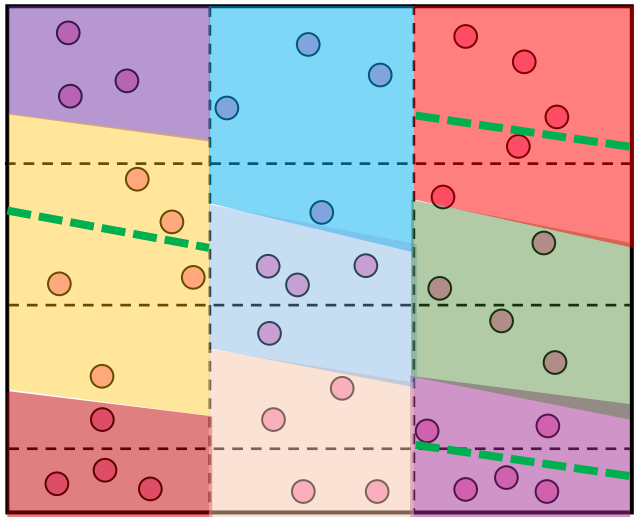
Building of $\mathcal{SP}$

- A preliminary study of a small real dataset
  - All mapped values are sorted in a list.
  - We plot the mapped values and their positions in the list.
  - A single regression function is hard to fit the whole dataset
- Form of our $\mathcal{SP}$
  - A piecewise linear regression function
- More formally
  - We need to find a list of numbers $\boldsymbol{M}_p = [\tilde{m}_1, \ldots, \tilde{m}_U]$ that evenly partition the mapped values.
  - For each interval $[\tilde{m}_{i-1}, \tilde{m}_i)$, we learn a regression model $\mathcal{F}_i$.



Step 4: Shards, Pages and Local Models

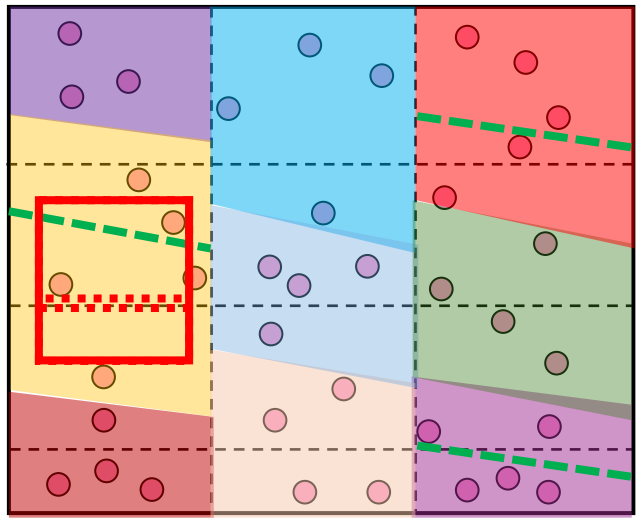
- A point $\boldsymbol{x}$ is assigned to a shard based on $\mathcal{SP}(\mathcal{M}(\boldsymbol{x}))$.
  - A shard may contain too many points for a disk page.



- For each shard S_i, we use a local model $\mathcal{L}_i$ to manage its pages.
  - $\mathcal{L}_i$ partitions the mapped values in S_i into a number of pages.
    - A list of pages
    - A list of partitioning mapped values
- Shard and page: One-to-many

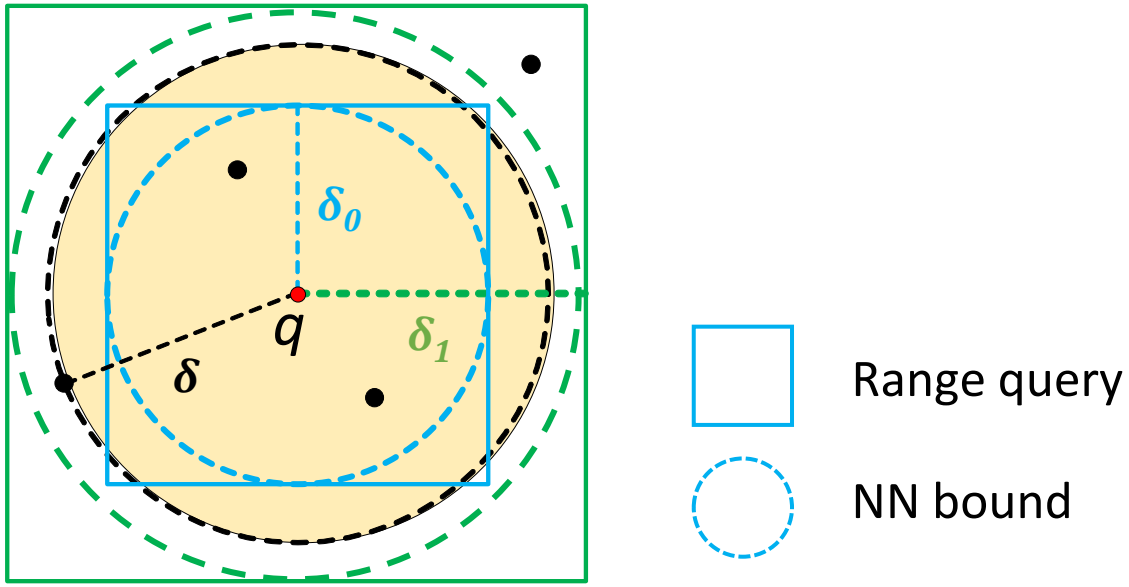
Range Query via LISA

- Given a query rectangle *qr*
  - *W.r.t.* the grid, decompose *qr* into *Q* sub-rectangles each in a grid cell
  - For each sub-rectangle, plug its lower and upper corners into $\mathcal{M}(.)$ and $\mathcal{SP}(.)$ to obain the shard
  - Use the shard's local model to find the relevant disk pages
    - Refinement: For each disk page, add all points falling in *qr* to the query result.
- LISA accesses relevant cells, shards and pages only



KNN Query via LISA

- Given a KNN query (q, K) to find q's K nearest neighbors
  - Convert the KNN query into a series of range queries
    - Start with an initial range δ_0
    - Use a lattice regression model to estimate δ_0 for a given query point and K
  - Expand the query range by a ratio > 1, until there're enough points found
- In this example
  - k = 3



Updates of LISA

- Insertion of point $\boldsymbol{x}$
  - Calculate the mapped value: $\mathcal{M}(\boldsymbol{x})$
  - Get the shard id: $i = \mathcal{SP}(\mathcal{M}(\boldsymbol{x}))$
  - Use the local model $\mathcal{L}_i$ to get the page
    - If the page has space, insert $\boldsymbol{x}$ into the page
    - Otherwise, split the page, insert $\boldsymbol{x}$ and update $\mathcal{L}_i$
- Deletion of point $\boldsymbol{x}$
  - Calculate the mapped value: $\mathcal{M}(\boldsymbol{x})$
  - Get the shard id: $i = \mathcal{SP}(\mathcal{M}(\boldsymbol{x}))$
  - Use the local model $\mathcal{L}_i$ to get the page
    - Delete the point $\boldsymbol{x}$ from the page
    - Merge two continuous pages if needed, e.g., one of them is empty

Updates only affect a local model

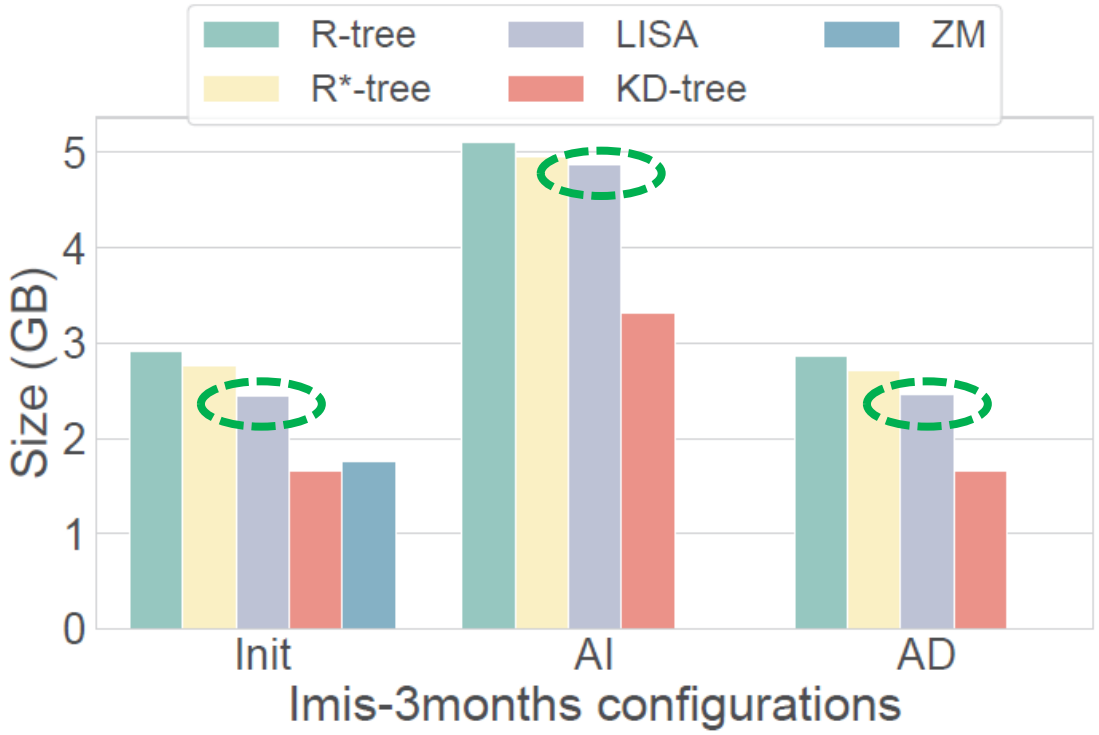
Experimental Settings

- Real and synthetic datasets
  - Imis-3months (2d): 98M AIS data points
  - ImageNet (6d): 73M points
  - Zipf (2d-6d): 100M points
  - Uniform (2d-6d): 100M points
- Different competitors
  - R-tree, R*-tree, and KD-tree
  - ZM
- Different configurations
  - **Init**: Building models using 50% of the data
  - **AI**: Inserting 50% of the data on the basis of 'Init'
  - **AD**: Random deleting 50% of the data on the basis of 'AI'

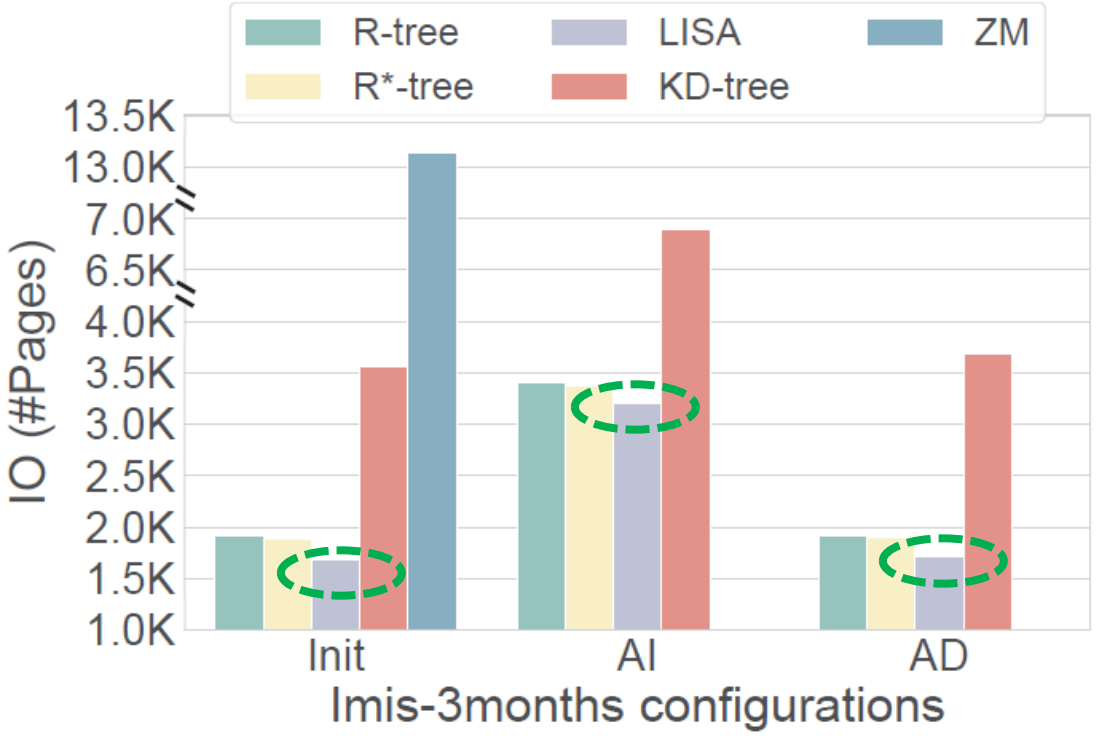
Parameter	**Setting**
Disk page size (PS)	4096 bytes
MBR size (MS)	$8 \times 2 \times d = 16d$ bytes
Page address size (AS)	4 bytes
#Keys in a page	$\frac{\text{PS}}{\text{MS} + \text{AS}}$

Performance Results: All Configurations

- Index sizes



- IO costs of range queries



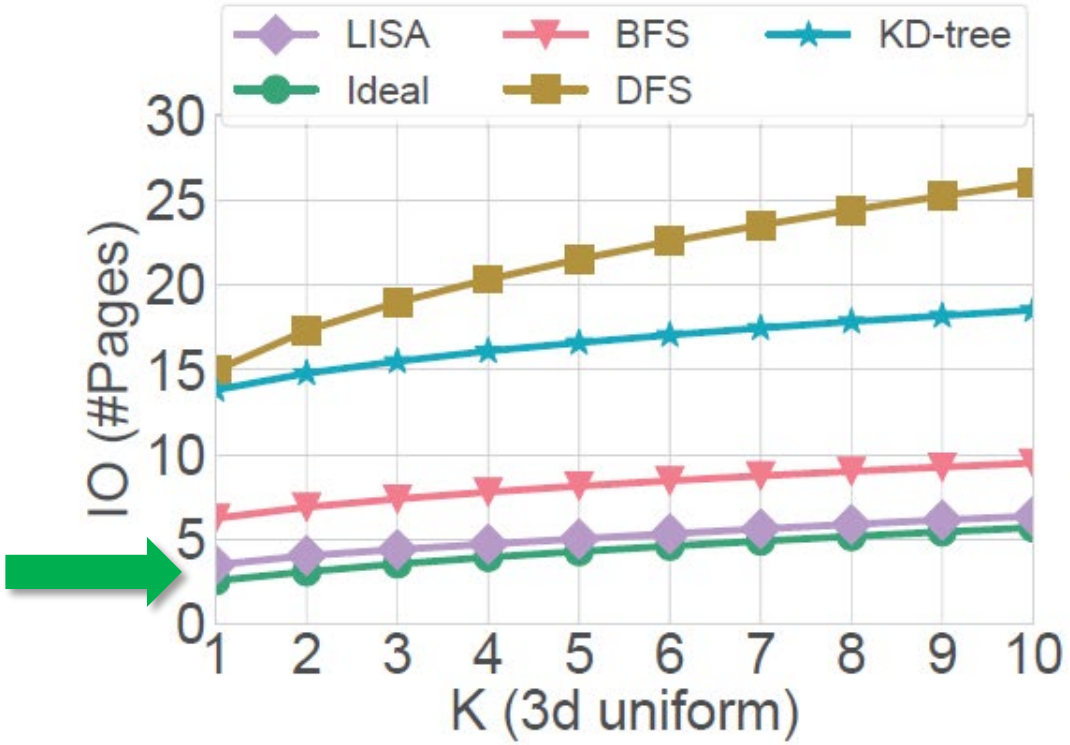
Range Query Time Costs (Init Configuration)

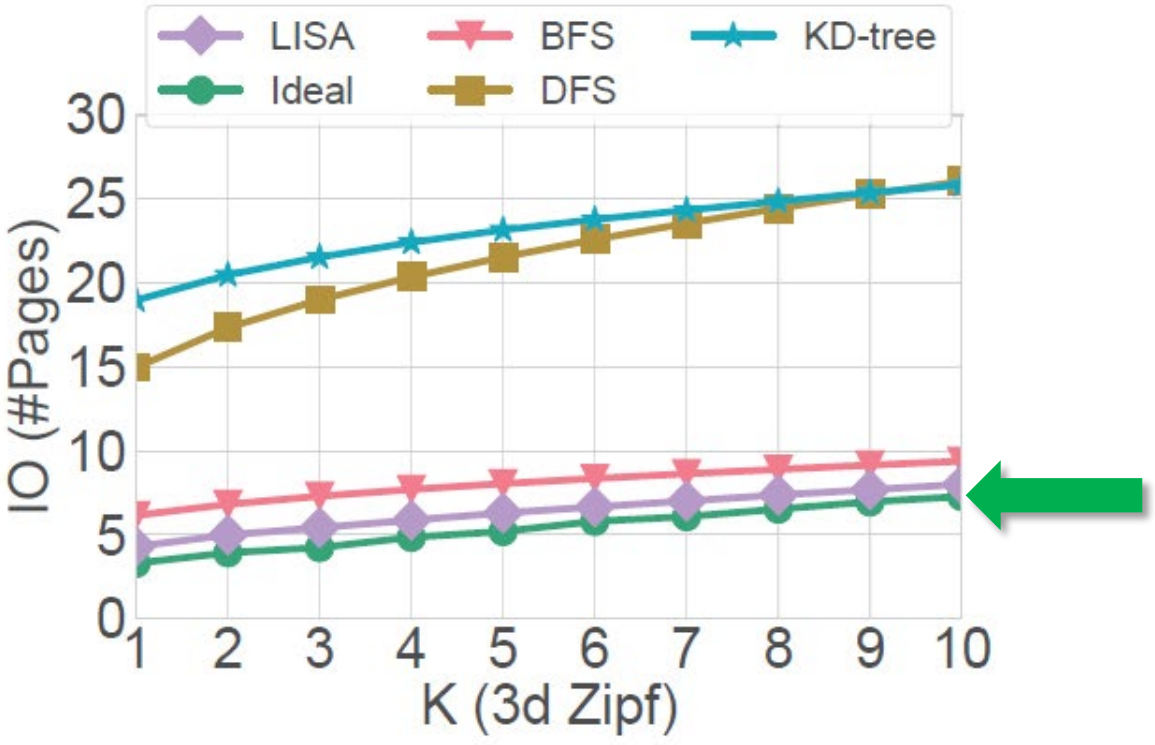
- CPU time and response time
  - Average of 10,000 range queries
  - LISA's response time is the shortest.

Method	3D Uniform		ImageNet	
	CPU time (ms)	Response time (ms)	CPU time (ms)	Response time (ms)
R-tree	1.34	11.26	3.85	39.50
R*-tree	1.31	11.08	3.61	37.79
KD-tree	729	765.5	5,655	6,378
ZM	1.97	246.4	2.32	1,173
LISA	1.43	**10.07**	5.08	**27.22**

KNN Query Performance

- LISA is very close to the **Ideal** case
  - ‘Ideal’ uses the real K-th NN’s distance as the range for KNN via LISA.





Summary of Learned Indexes

- RMI is the first learned index for 1D keys.
- ZM is a simplistic learned index for spatial points.
  - Z-order + RMI
- LISA's design is more sophisticated, and thus is more effective than traditional indexes and ZM.
  - Grid cells, shards, and pages

	RMI	**ZM**	**LISA**
Data	1D keys	Multi-D points	Multi-D points
Model	NN + Linear models	NN	Piecewise linear model
Updates	✗	✗	✓

Agenda

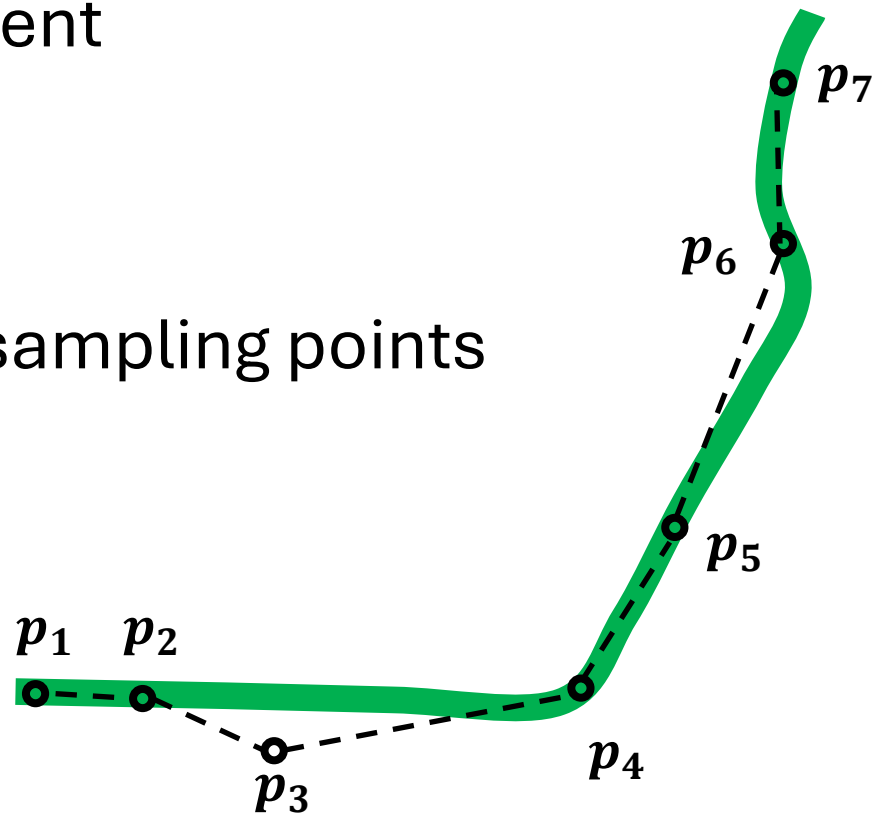
- Introduction
- Learned index for spatial data
- Representation learning of spatial data
  - CLEAR: Representation learning of trajectories
  - Poly2Vec: Unified representation learning of spatial objects
- Machine learning for location-based services
- Conclusion and future research

Agenda

- Introduction
- Learned index for spatial data
- Representation learning of spatial data
  - CLEAR: Representation learning of trajectories
  - Poly2Vec: Unified representation learning of spatial objects
- Machine learning for location-based services
- Conclusion and future research

Characteristics of Trajectory Data

- Discretization of continuous movement
- Low sampling rates
- Irregular sampling rates
- Varied length between consecutive sampling points
  - Both spatial and temporal lengths
- Noises



$$T = (p_1, p_2, p_3, p_4, p_5, p_6, p_7)$$

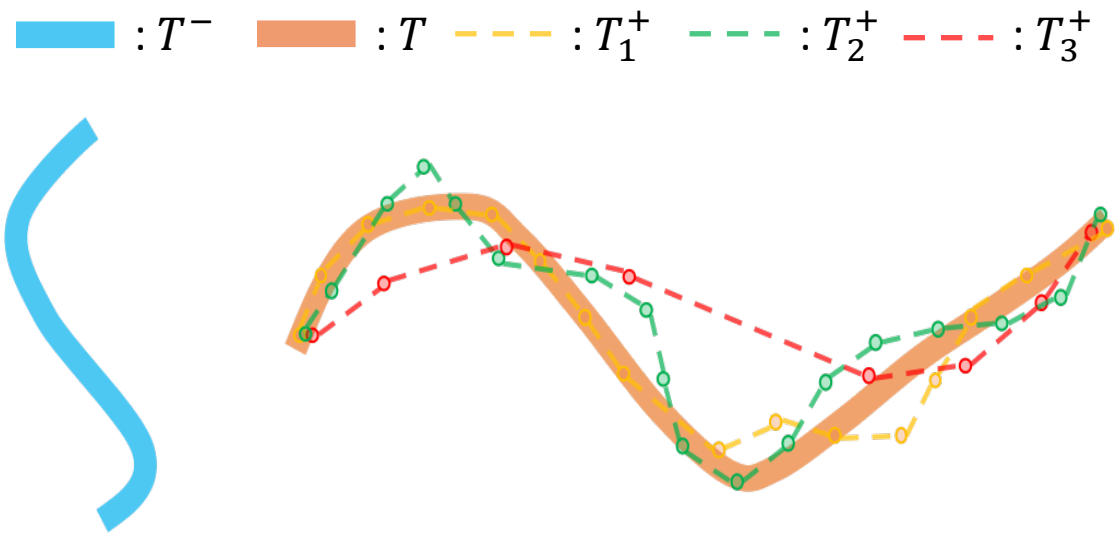
Trajectory Similarity

- Intuition: $sim(T, T_1^+) > sim(T, T_2^+) > sim(T, T_3^+) \gg sim(T, T^-)$

Easy positive

Hard positive

- Objective
  - Robust trajectory similarity computation that can handle easy and hard positives well



Existing Approaches

- **Heuristic Methods**
  - EDwP
  - Fréchet
  - Hausdorff
  - …
- Drawbacks
  - Inefficient point matching
  - Sensitive to data uncertainties

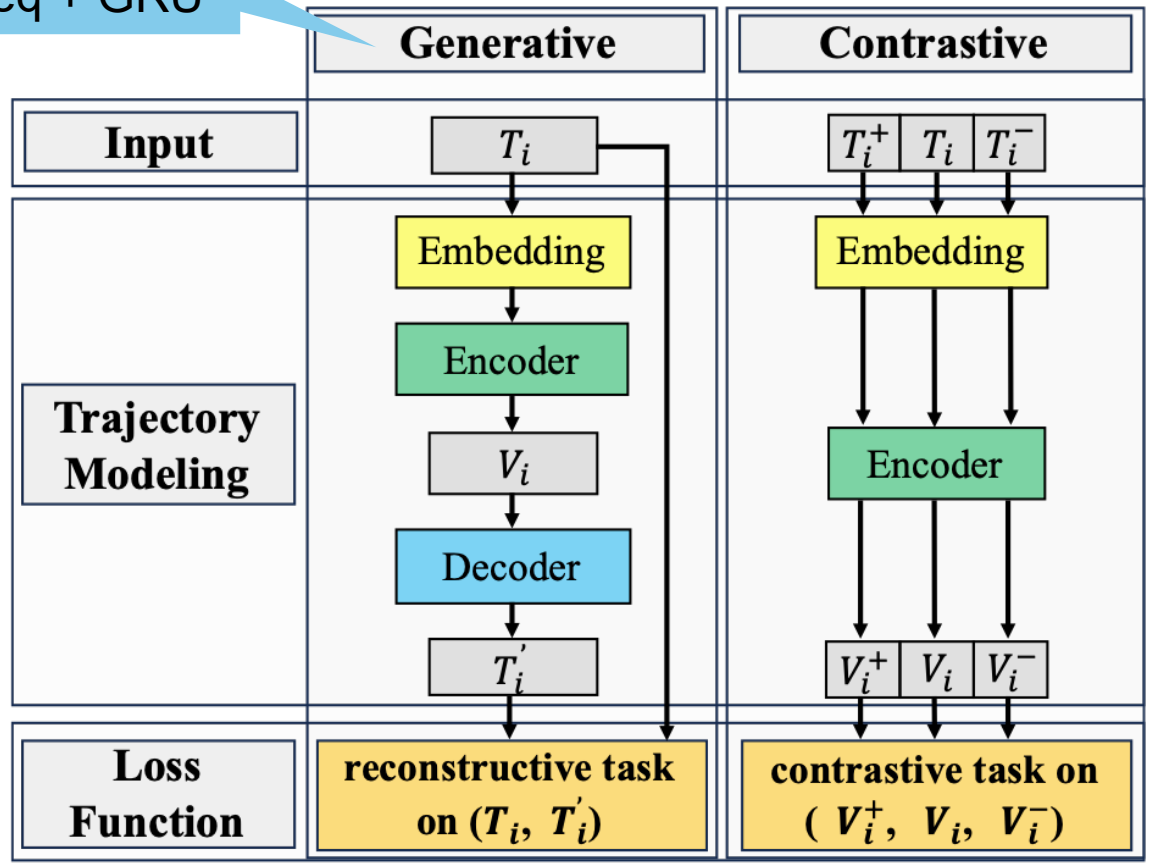
- **Learned Models**
  - Trajectory embedding
  - Embedding similarity as trajectory similarity

$$sim(T_i, Tj) \Rightarrow sim(V_i, V_j)$$

- Pros and cons
  - Efficient: fixed-length vectors
  - Robust?
  - No ground truth label ☹

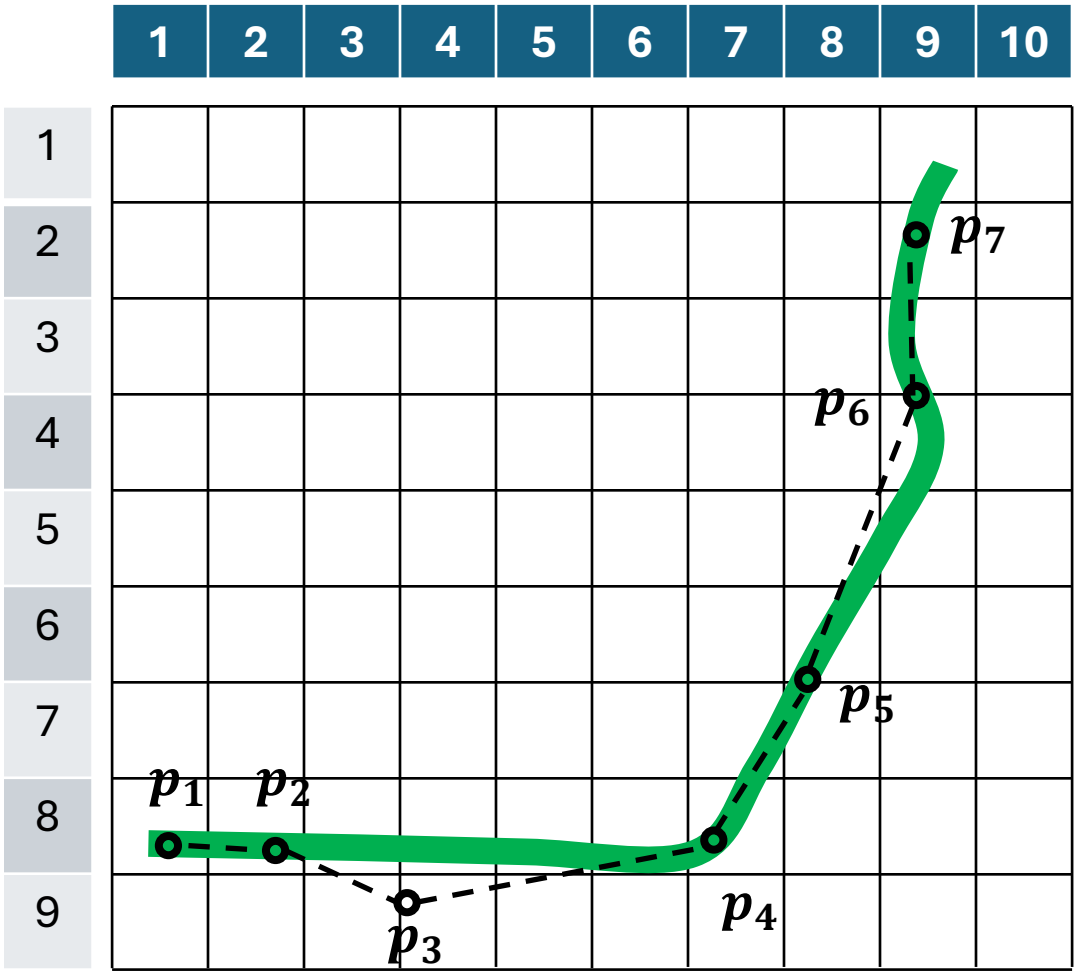
Generative vs. Contrastive Models

- **t2vec** [ICDE'18]: seq2seq + GRU



- **Contrastive models**
  - **CLTSim** [CIKM'22]: CL + LSTM
  - **TrajCL** [ICDE'23]: CL + Transformer
- Drawbacks
  - Inadequate modeling
    - CLTSim: sequential only
    - TrajCL: spatial only
  - Insufficient handling of multi-positive
  - One pair of data augmentation only
- Our design: **C**ontrastive representation **lear**ning (**CLEAR**)

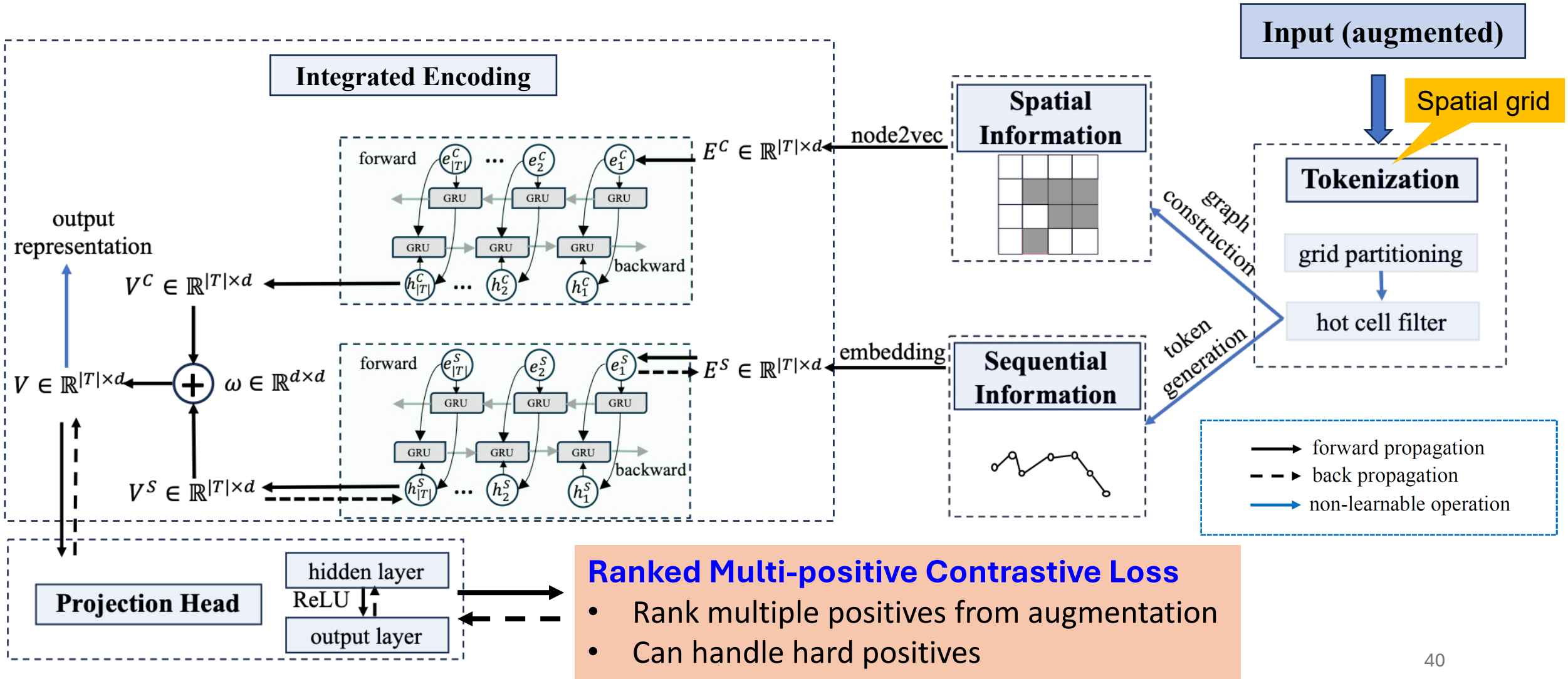
Tokenization



- Sampling points -> cell ID
- In this example
  - $T = \langle p_1, p_2, p_3, p_4, p_5, p_6, p_7 \rangle$
  → ⟨71, 72, 84, 77, 58, 39, 19⟩
- Benefits
  - Smaller learned space
  - Adjustable resolution

Dataset	# Trajectories	# Cells
Porto	1,227,467	23,751
GeoLife	69,758	37,106

CLEAR: Trajectory Modeling & Contrastive Loss



Ranked Multi-positive Contrastive Loss

- Rank multiple positives from augmentation
- Can handle hard positives

Ranked Multi-positive Loss

InfoNCE

1 positive

$$\mathcal{L}_{NCE} = -log \frac{\exp(\frac{V_i \cdot V_j}{\tau})}{\sum_{k=1}^{2n} \mathbb{1}_{[k \neq i]} \exp(\frac{V_i \cdot V_k}{\tau})}$$

SupCon

m positive

$$\mathcal{L}_{out} = \frac{1}{|\mathcal{P}(i)|} \sum_{j \in \mathcal{P}(i)} -log \frac{\exp(\frac{V_i \cdot V_j}{\tau})}{\exp\left(\frac{V_i \cdot V_j}{\tau}\right) + \sum_{k \in \mathcal{N}(i)} exp\left(\frac{V_i \cdot V_k}{\tau}\right)}$$

CLEAR

hard positive

$$\mathcal{L}_{rank} = \frac{1}{(m-1)} \sum_{\mathcal{R}=1}^{m-1} \frac{1}{|\mathcal{P}_{\mathcal{R}}(i)|} \sum_{j \in \mathcal{P}_{\mathcal{R}}(i)} -log \frac{\exp(\frac{V_i \cdot V_j}{\tau})}{\sum_{j \in \mathcal{P}_{\mathcal{R}}(i)} exp\left(\frac{V_i \cdot V_j}{\tau}\right) + \sum_{k \in \mathcal{N}(i)} exp\left(\frac{V_i \cdot V_k}{\tau}\right)}$$

How to generate multi-positives? Data augmentation

CLEAR: Multiple Data Augmentation

- Distortion: We move a sampling point
- Downsampling: We discard a sampling point

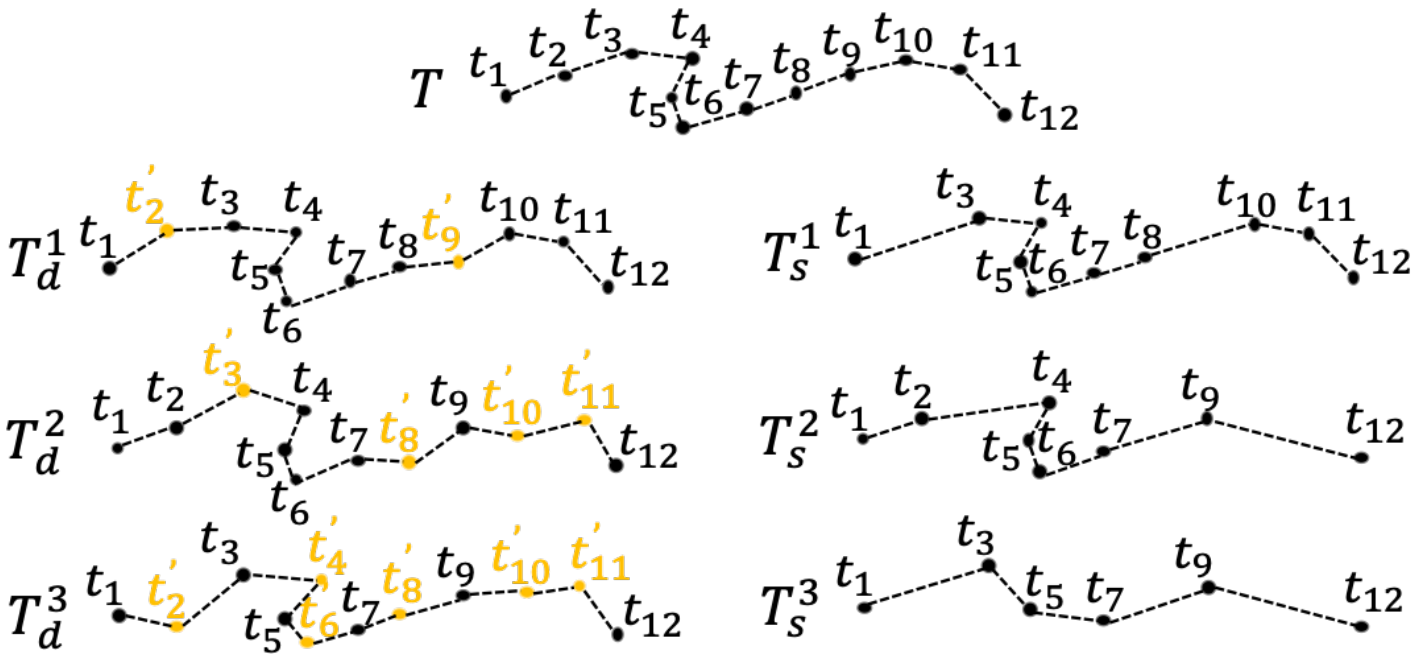
T original trajectory

T_d^i distorted T

T_s^i downsampled T

• sampled points

• distorted points



Experimental Results: kNN Search

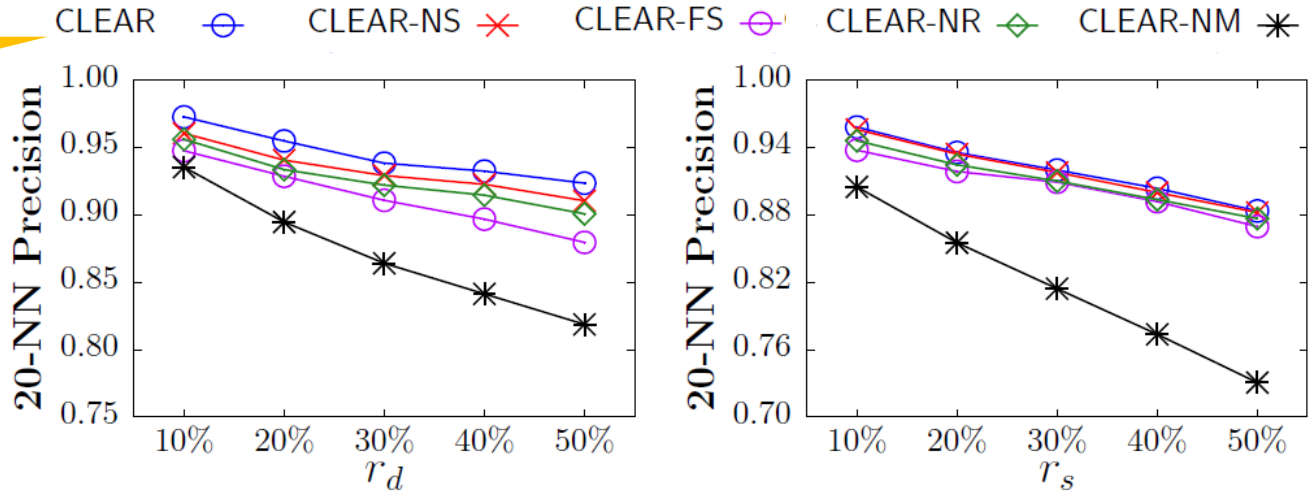
- Metric: $|R_V \cap R_T| / k$
  - kNN in embedding space (R_V) *vs.* KNN in original trajectories (R_T)

Porto		Distortion (r_d)					Downsampling (r_s)				
k	Model	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
20	t2vec	0.8508	0.7810	0.7261	0.6889	0.6448	0.8492	0.7830	0.7333	0.6752	0.6268
	CLTSim	0.9202	0.8827	0.8384	0.8087	0.7798	0.8904	0.8334	0.7889	0.7456	0.6958
	TrajCL	**0.9778**	**0.9639**	**0.9456**	**0.9350**	0.9114	0.7542	0.6762	0.6297	0.5998	0.5858
	CLEAR	0.9728	0.9550	0.9385	0.9326	**0.9236**	**0.9583**	**0.9356**	**0.9200**	**0.9038**	**0.8836**
30	t2vec	0.8618	0.7996	0.7483	0.7057	0.6630	0.8625	0.7968	0.7496	0.6941	0.6479
	CLTSim	0.9283	0.8875	0.8449	0.8243	0.7934	0.8950	0.8462	0.8023	0.7578	0.7103
	TrajCL	**0.9753**	**0.9612**	**0.9459**	0.9385	0.9285	0.7612	0.6812	0.6307	0.6008	0.5918
	CLEAR	0.9751	0.9609	0.9457	**0.9388**	**0.9291**	**0.9612**	**0.9412**	**0.9270**	**0.9130**	**0.8963**
40	t2vec	0.8692	0.8098	0.7590	0.7178	0.6795	0.8708	0.8080	0.87606	0.7096	0.6634
	CLTSim	0.9321	0.8904	0.8502	0.8279	0.8009	0.8997	0.8498	0.8063	0.7636	0.7206
	TrajCL	0.9753	0.9612	0.9459	0.9390	0.9285	0.7652	0.6851	0.6352	0.6108	0.5958
	CLEAR	**0.9777**	**0.9625**	**0.9494**	**0.9432**	**0.9337**	**0.9645**	**0.9460**	**0.9303**	**0.9172**	**0.9010**

Experimental Results: Ablation Study

- **CLEAR-NS** drops out the *spatial embedding of cells*
- **CLEAR-FS** fuses the spatial and sequential embeddings *before* encoding
- **CLEAR-NR** considers multiple positives but *no hard positives* in the contrastive loss
- **CLEAR-NM** considers *only one positive* in the contrastive loss

Our CLEAR is more robust

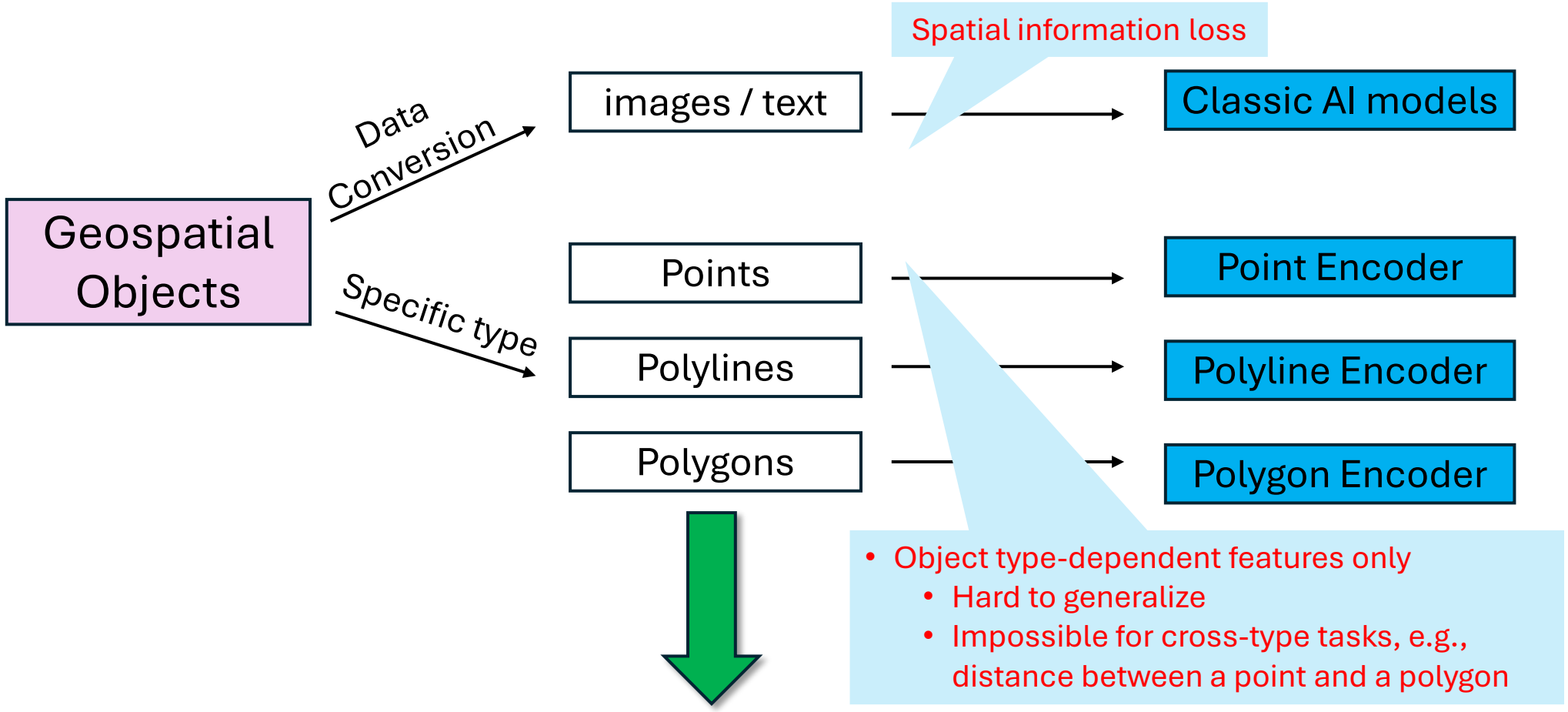


(a) Distortion (b) Downsampling

Agenda

- Introduction
- Learned index for spatial data
- Representation learning of spatial data
  - CLEAR: Representation learning of trajectories
  - Poly2Vec: Unified representation learning of spatial objects (*Joint work with USC's InfoLab*)
- Machine learning for location-based services
- Conclusion and future research

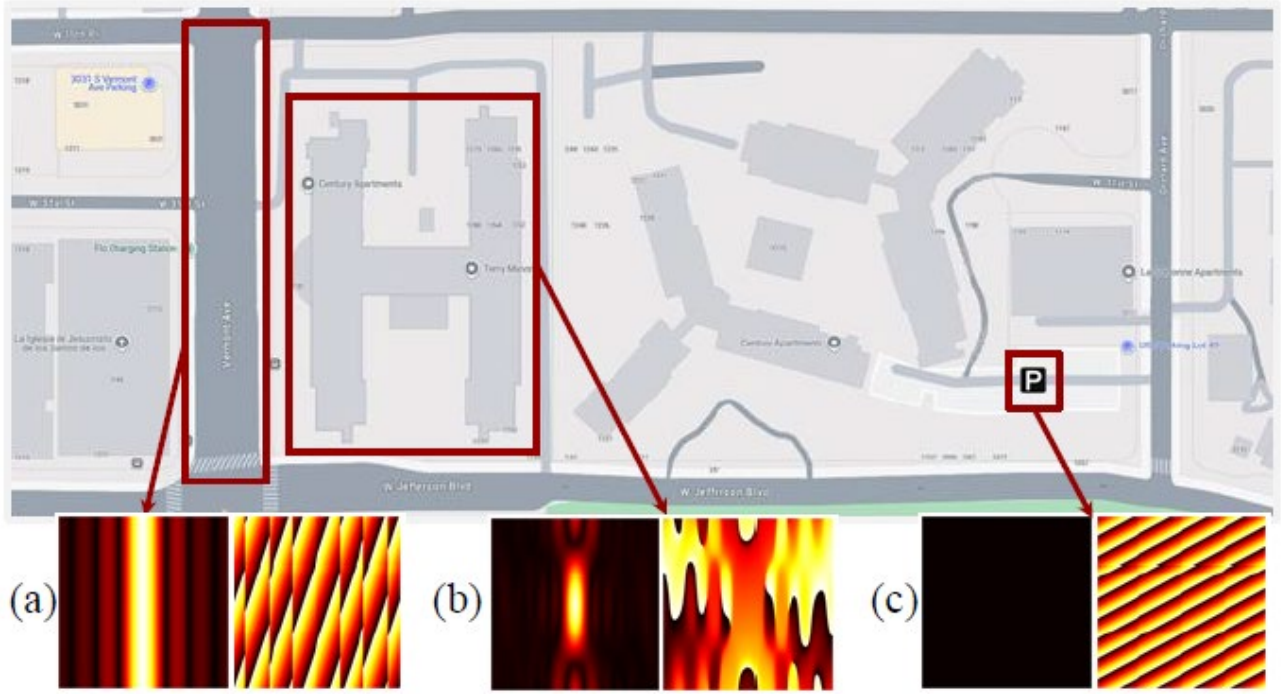
Motivation



Poly2Vec: Unifying the geometry encoding while preserving spatial properties

Visualization of Fourier Transform

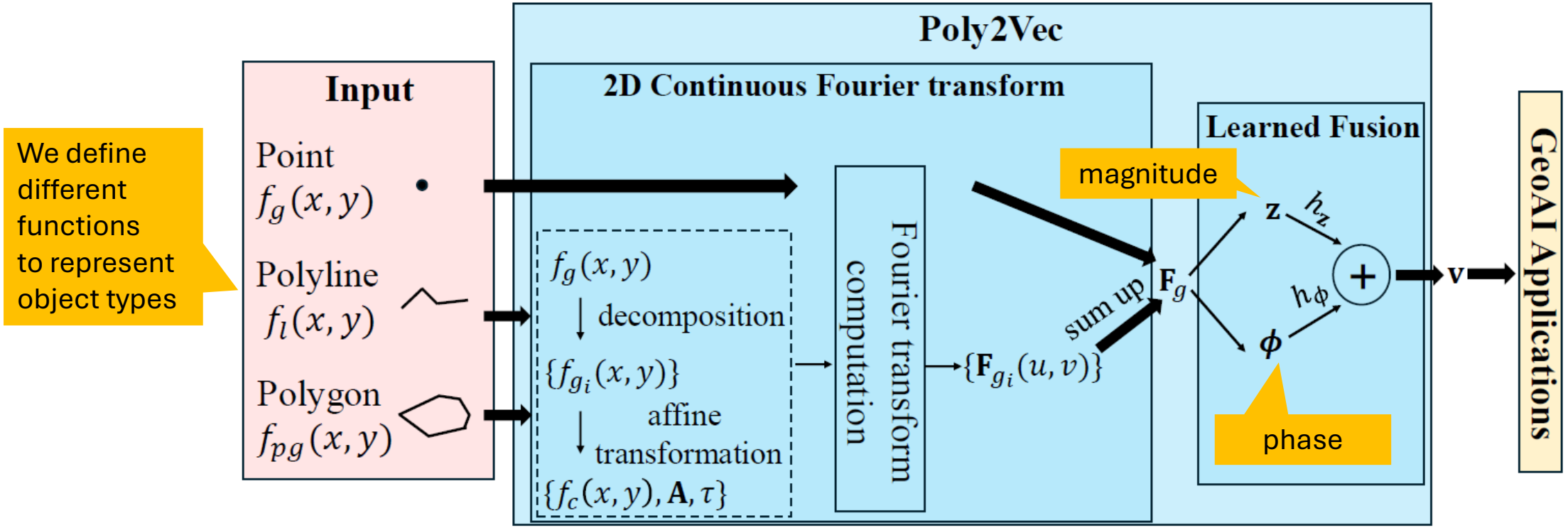
- Example
  a) Road segment
  b) Building
  c) POI



- Magnitude and phase complement each other
  - Magnitude reflects spatial extent, being uniform for points with no shape and varying for polygons and polylines
  - Phase highlights directionality, such as the alignment of a polyline.

Workflow of Poly2Vec

- Unifying encoding based on Fourier transform's **Linearity** and **Affine transformation**.



Experimental Study

- OSM data from two cities: **Singapore** and **New York**
  - POIs -> Points
  - Main roads -> Polylines
  - Buildings -> Polygons
- Spatial reasoning
  - Topological relationship (*classification*)
  - Directional relationship (*classification*)
  - Distance estimation (*regression*)

Results: Topological Relationship

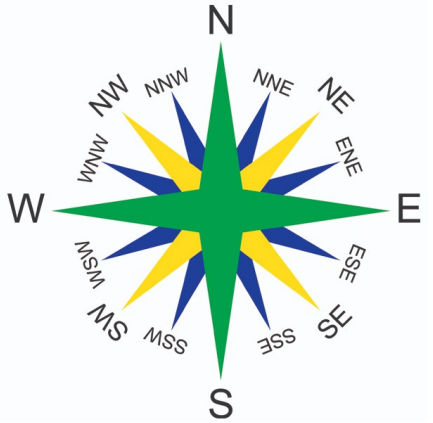
- The geometry embeddings of a pair are concatenated, passed through a 2-layer MLP classifier

Geometry Pair	Topological Relationships (a relationship b)
point-polyline	disjoint, intersects
point-polygon	disjoint, contains
polyline-polyline	disjoint, intersects
polyline-polygon	disjoint, touches, intersects, within
polygon-polygon	disjoint, touches, intersects, contains, within, equals

Methods	Singapore					New York				
	point-polyline	point-polygon	polyline-polyline	polyline-polygon	polygon-polygon	point-polyline	point-polygon	polyline-polyline	polyline-polygon	polygon-polygon
ResNet1D	-	-	-	-	$0.457_{0.017}$	-	-	-	-	$0.452_{0.033}$
NUFTspec	-	-	-	-	$\underline{0.602}_{0.009}$	-	-	-	-	$\underline{0.585}_{0.008}$
t2vec	-	-	$\underline{0.728}_{0.023}$	-	-	-	-	$\underline{0.807}_{0.121}$	-	-
Direct	$0.823_{0.013}$	$0.843_{0.005}$	$0.733_{0.007}$	$0.368_{0.010}$	$0.357_{0.018}$	$0.846_{0.011}$	$0.909_{0.018}$	$0.745_{0.008}$	$0.495_{0.009}$	$0.446_{0.023}$
Tile	$0.790_{0.021}$	$0.700_{0.010}$	$0.505_{0.005}$	$0.459_{0.013}$	$0.411_{0.009}$	$0.659_{0.013}$	$0.783_{0.007}$	$0.502_{0.009}$	$0.494_{0.038}$	$0.405_{0.005}$
Wrap	$0.886_{0.003}$	$0.880_{0.008}$	$0.716_{0.011}$	$\underline{0.476}_{0.010}$	$0.349_{0.004}$	$0.886_{0.006}$	$0.880_{0.017}$	$0.733_{0.009}$	$0.550_{0.011}$	$0.381_{0.007}$
Grid	$0.846_{0.004}$	$0.844_{0.004}$	$0.697_{0.031}$	$0.458_{0.004}$	$0.335_{0.012}$	$0.822_{0.039}$	$0.891_{0.004}$	$0.739_{0.009}$	$0.516_{0.008}$	$0.381_{0.031}$
Theory	$\underline{0.892}_{0.003}$	$\underline{0.900}_{0.005}$	$0.719_{0.008}$	$0.450_{0.010}$	$0.461_{0.041}$	$\underline{0.897}_{0.008}$	$\underline{0.909}_{0.008}$	$0.734_{0.008}$	$\underline{0.591}_{0.006}$	$0.455_{0.041}$
Poly2Vec	$\mathbf{0.955}_{0.007}$	$\mathbf{0.949}_{0.002}$	$\mathbf{0.812}_{0.010}$	$\mathbf{0.509}_{0.008}$	$\mathbf{0.702}_{0.006}$	$\mathbf{0.953}_{0.003}$	$\mathbf{0.980}_{0.002}$	$\mathbf{0.830}_{0.004}$	$\mathbf{0.641}_{0.062}$	$\mathbf{0.684}_{0.008}$

Results: Directional Relationship

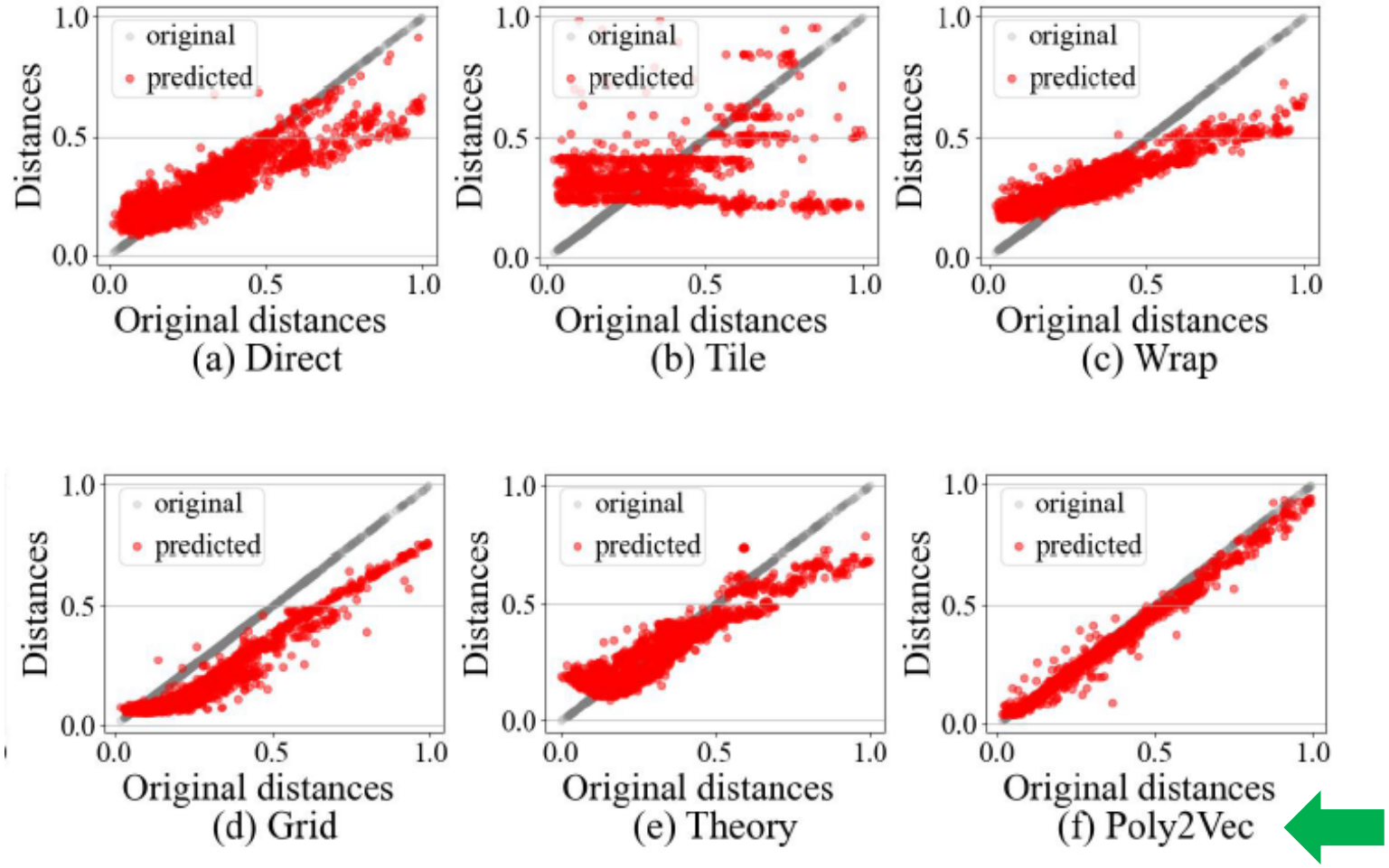
- 16-compass model of two geospatial objects
- Again, a 2-layer MLP classifier is used



Methods	Singapore						New York					
	point-point	point-polyline	point-polygon	polyline-polyline	polyline-polygon	polygon-polygon	point-point	point-polyline	point-polygon	polyline-polyline	polyline-polygon	polygon-polygon
ResNet1D	-	-	-	-	-	0.819$_{0.010}$	-	-	-	-	-	0.747$_{0.010}$
NUFTspec	-	-	-	-	-	0.807$_{0.008}$	-	-	-	-	-	0.698$_{0.017}$
t2vec	-	-	-	0.268$_{0.075}$	-	-	-	-	-	0.249$_{0.032}$	-	-
Direct	0.880$_{0.006}$	0.841$_{0.007}$	0.844$_{0.006}$	0.820$_{0.002}$	0.830$_{0.005}$	0.752$_{0.017}$	0.877$_{0.004}$	0.766$_{0.005}$	0.836$_{0.008}$	0.653$_{0.007}$	0.784$_{0.004}$	0.694$_{0.004}$
Tile	0.253$_{0.001}$	0.268$_{0.002}$	0.273$_{0.008}$	0.326$_{0.010}$	0.454$_{0.001}$	0.394$_{0.003}$	0.245$_{0.009}$	0.258$_{0.005}$	0.316$_{0.005}$	0.217$_{0.001}$	0.466$_{0.001}$	0.349$_{0.012}$
Wrap	0.861$_{0.018}$	0.804$_{0.009}$	0.803$_{0.004}$	0.781$_{0.002}$	0.831$_{0.002}$	0.778$_{0.001}$	0.809$_{0.004}$	0.669$_{0.001}$	0.749$_{0.018}$	0.596$_{0.019}$	0.772$_{0.002}$	0.602$_{0.006}$
Grid	0.882$_{0.007}$	0.728$_{0.007}$	0.771$_{0.003}$	0.699$_{0.001}$	0.641$_{0.016}$	0.534$_{0.138}$	0.868$_{0.002}$	0.590$_{0.003}$	0.646$_{0.049}$	0.438$_{0.004}$	0.752$_{0.001}$	0.485$_{0.079}$
Theory	0.912$_{0.014}$	0.867$_{0.009}$	0.858$_{0.004}$	0.834$_{0.012}$	0.860$_{0.006}$	0.735$_{0.044}$	0.892$_{0.017}$	0.760$_{0.007}$	0.826$_{0.008}$	0.684$_{0.009}$	0.775$_{0.005}$	0.555$_{0.012}$
Poly2Vec	**0.932$_{0.006}$**	**0.935$_{0.032}$**	**0.925$_{0.002}$**	**0.906$_{0.010}$**	**0.907$_{0.007}$**	**0.833$_{0.006}$**	**0.909$_{0.012}$**	**0.891$_{0.004}$**	**0.883$_{0.013}$**	**0.863$_{0.007}$**	**0.876$_{0.009}$**	**0.785$_{0.003}$**

Results: Distance estimation

- The original distance is estimated by the Euclidean distance of the geometry embeddings



Agenda

- Introduction
- Learned index for spatial data
- Representation learning of spatial data
- Machine learning for location-based services
  - Data imputation for indoor positioning
  - Indoor population modeling and monitoring
- Conclusion and future research

Agenda

- Introduction
- Learned index for spatial data
- Representation learning of spatial data
- Machine learning for location-based services
  - Data imputation for indoor positioning
  - Indoor population modeling and monitoring
- Conclusion and future research

Why Indoor?

- People in need of information
- Complex indoor spaces (*topology*)
- Smartphones with Wi-Fi, Bluetooth, and other sensors (*positioning*)

Last-mile LBS

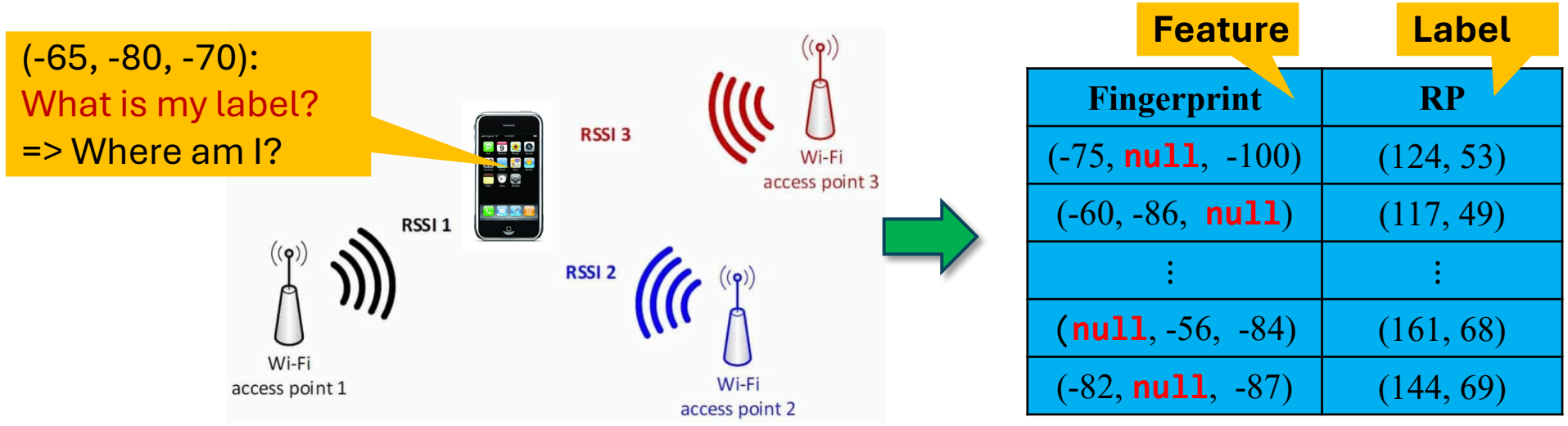
- People spend **87%** of time indoors
- Beijing Capital Airport
  - 260,000+ outbound passengers daily in 2017
- New Town Plaza, Hong Kong
  - 200,000 m^2, 34 interconnected buildings
  - Weekend traffic 320,000 people (2004)
- Copenhagen Airport
  - 2.38+ million passengers in March 2018
  - The busiest in its 92-year history

General Challenges of Indoor Space

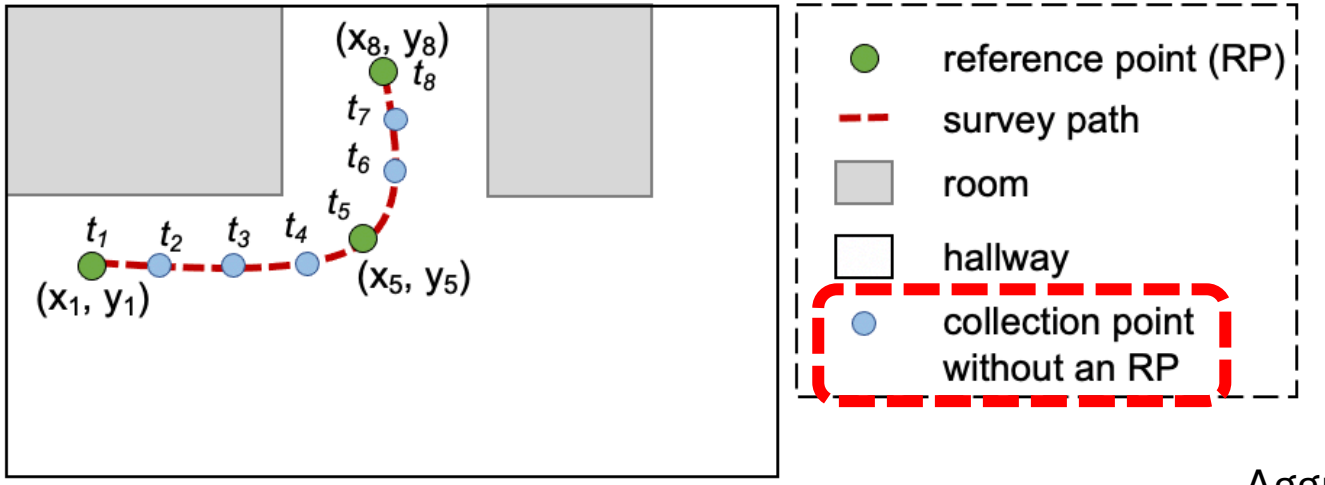
- Imprecise indoor positioning
  - GPS unavailable or unusable
  - Short-range wireless technologies such as WiFi, Bluetooth
    - Proximity analysis (an object is within the detection range of a sensor)
    - Fingerprinting
- Unique indoor topology
  - Rooms, doors, hallways
  - Semi-constrained movements
    - Unlike road networks
    - Not free-moving space either

Wi-Fi based Indoor Positioning (Fingerprinting)

- **Offline phase**: Collecting fingerprints at *reference positions* (RP) into radio map
  - **Fingerprint**: A vector of Wi-Fi *received signal strength indicator* values (RSSIs)
  - **Radio map**: Pairs of a fingerprint and a corresponding RP where the former is collected
- **Online phase**: Radio map + current fingerprint => user's current location
  - Location estimation as classification: NN, KNN, Weighted KNN, Random Forest (RF)
- **Problem**: Fingerprints have missing values in both phases



Walking-survey Based Radio Map Creation



Time	Type	Measurement	Time	Type	Measurement
$t_1 = 0$	RP	(x_1, y_1)	$t_5 = 9$	RP	(x_5, y_5)
$t_2 = 1$	RSSI	$\langle r_1 : -70, r_2 : -83, r_3 : -76 \rangle$	$t_6 = 12$	RSSI	$\langle r_1 : -74, r_5 : -80 \rangle$
$t_3 = 3$	RSSI	$\langle r_1 : -71, r_3 : -78 \rangle$	$t_7 = 13$	RSSI	$\langle r_2 : -77, r_5 : -82 \rangle$
$t_4 = 8$	RSSI	$\langle r_3 : -80, r_4 : -68 \rangle$	$t_8 = 16$	RP	(x_8, y_8)

Walking Survey Record Table

Aggregate & match

Created **Sparse** Radio Map

No.	Radio Map Record	Time
1	$((-70, -83, -76, \texttt{null}, \texttt{null}), (x_1, y_1))$	t_2
2	$((-71, \texttt{null}, -78, \texttt{null}, \texttt{null}), \texttt{null})$	t_3
3	$((\texttt{null}, \texttt{null}, -80, -68, \texttt{null}), (x_5, y_5))$	t_4
4	$((-74, -77, \texttt{null}, \texttt{null}, -81), \texttt{null})$	t_6
5	$((\texttt{null}, \texttt{null}, \texttt{null}, \texttt{null}, \texttt{null}), (x_8, y_8))$	t_8

- **Sparsity of RP**: due to reducing the labor cost.
- **Sparsity of RSSI**: only part of APs can be actually observed at a location.
- Missing rates in real data
  - **85.6%** to **93.7%**

Data Imputation for Sparse Radio Maps

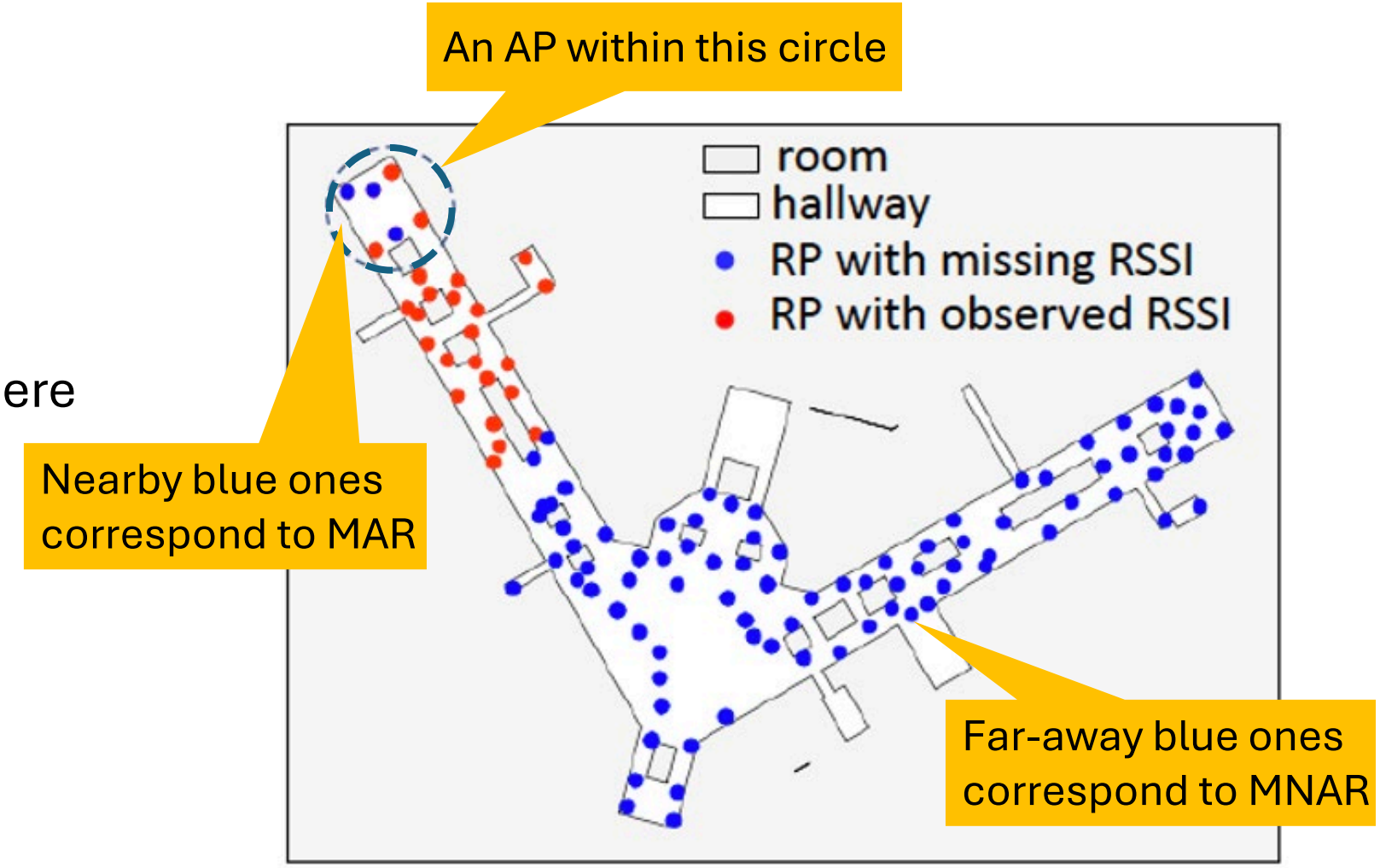
- Research Problem
  - To impute the missing RSSIs and missing RPs in the radio map such that indoor positioning using this radio map yields lower positioning errors.
- Co-existence of two kinds of missing RSSIs:
  - **Missing Not At Random (MNAR)**[1] RSSI: caused by weakness of signal---unobservable, usually filled-in with -100 dBm.
  - **Missing At Random (MAR)** RSSI: caused by random events such as occasional loss of contact with access points.
    - MARs are observable but unobserved.
    - Such values should be imputed.

[1] Little, Roderick J., and Donald B. Rubin. "Statistical analysis with missing data (Vol. 793)." (2019).

Example of MNAR and MAR RSSI

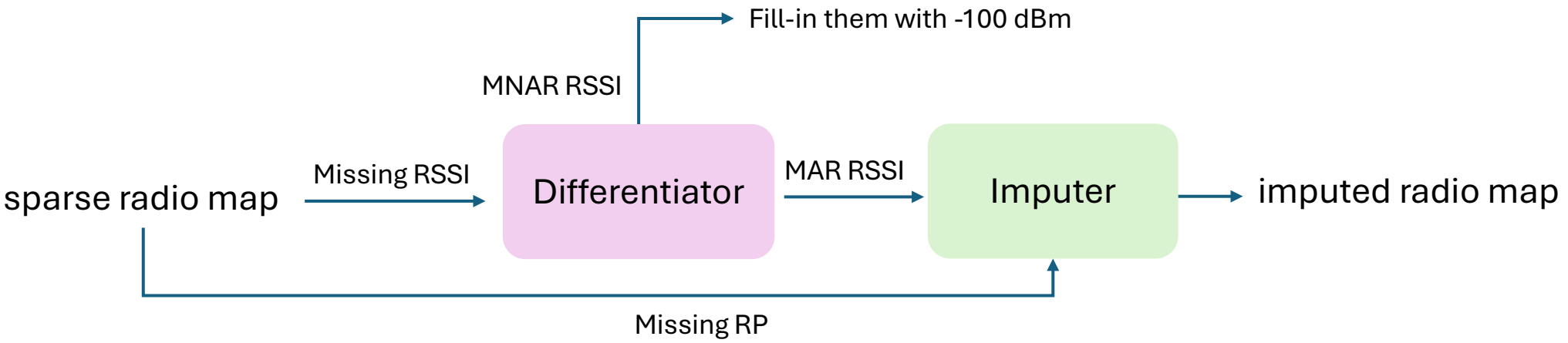
- Red RP
  - All fingerprints there observe the AP
- Blue RP
  - Some fingerprints there miss the AP

- It's only possible to impute MAR.



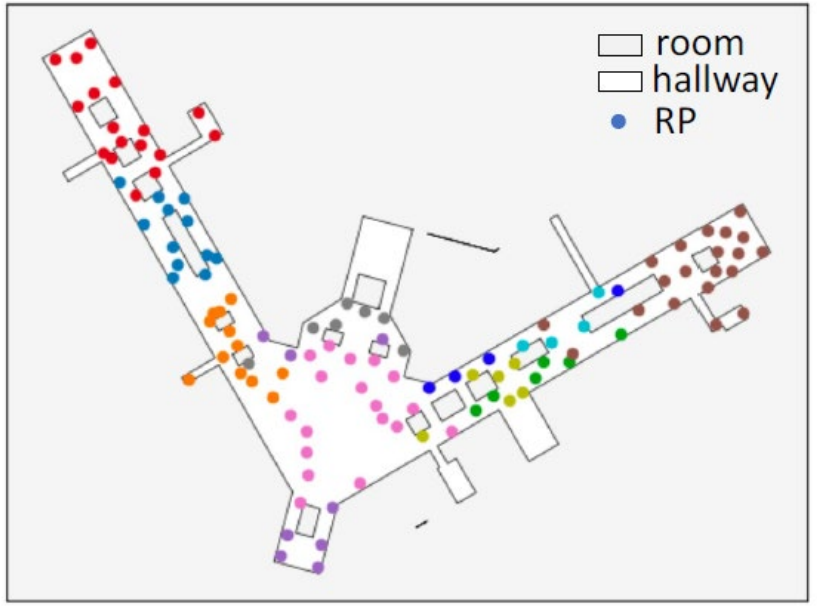
Solution Framework

- **Differentiator** differentiates MNAR RSSI and MAR RSSI
  - After differentiation, fill-in MNAR RSSI directly with -100 dBm
- **Imputer** imputes MAR RSSI and missing RP jointly



Differentiation of MNAR and MAR

- **Hypothesis**: Within a certain small range of space, the observability of APs is similar due to the similar signal transmission surroundings.
- To test the hypothesis
  1. RP's profile: A bitstring with each bit corresponding to an AP
     - 1 means the AP is observed at the RP
     - 0 otherwise.
  2. RP clustering based on their profiles
     - The hypothesis mostly holds.
     - Some exceptions occur due to noise (MAR)
- We identify MARs using the clusters.
  - They're outliers in each cluster.

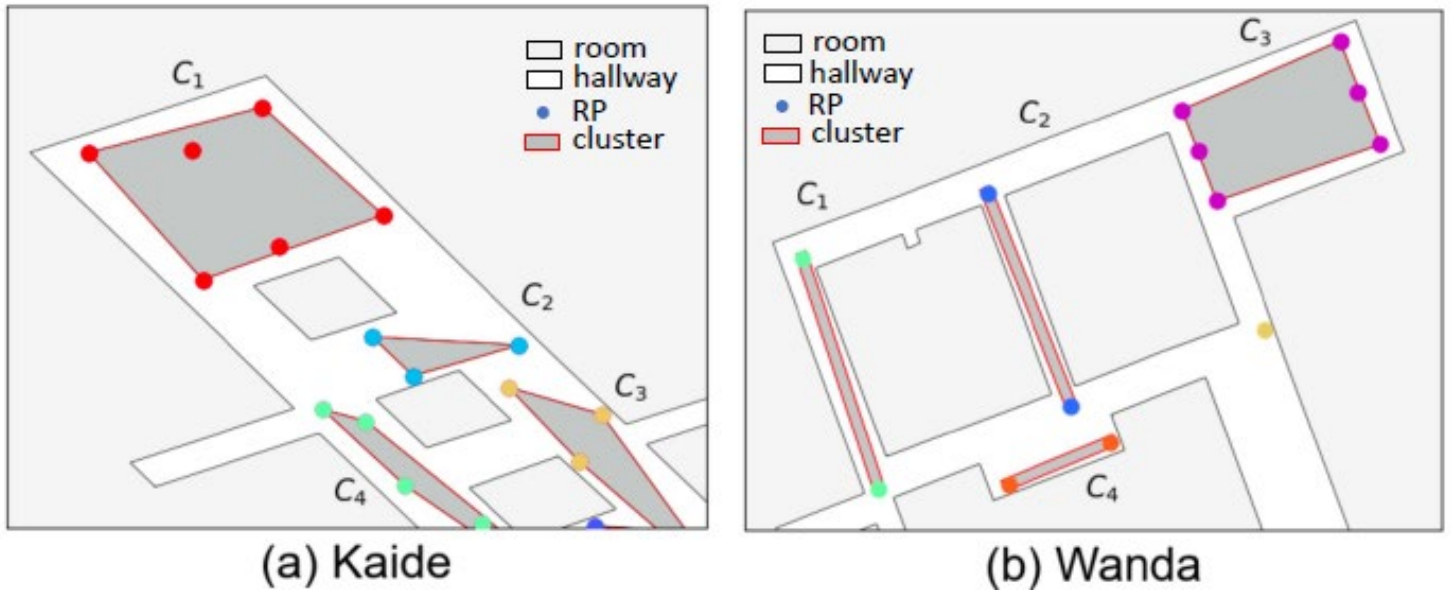


K-means Clustering for Differentiation

- Rough idea
  - Defining a metric of Differentiation Accuracy (DA)
  - Running a series K-means with different K values
    - Choosing the K value (and the clustering result) that maximizes DA
- Abnormal cases
  - This big cluster contains RPs at different rooms.
  - These RPs are dissimilar due to topology.
- We should consider topology in clustering!

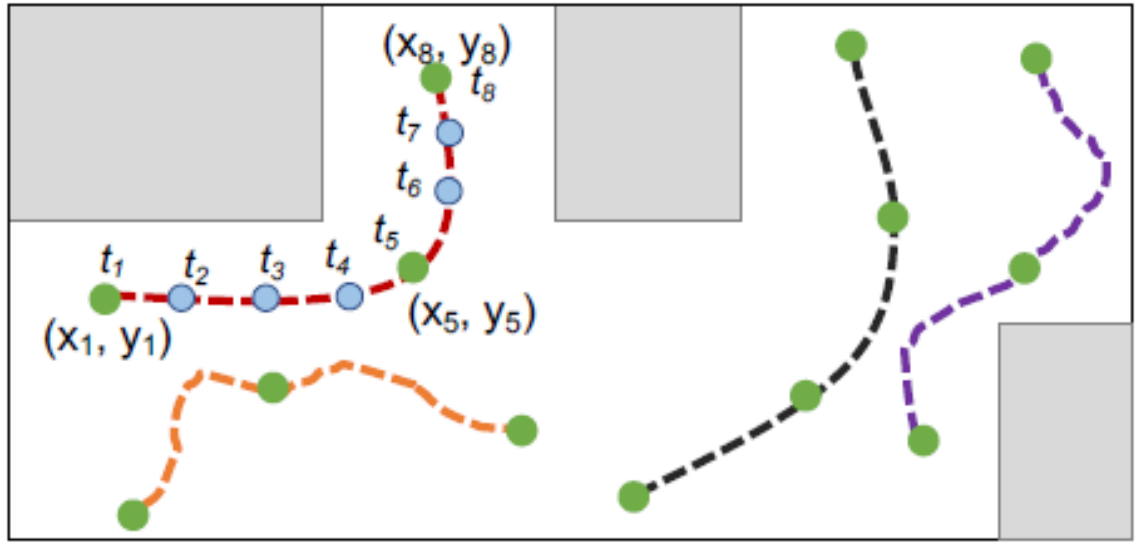
Topology-aware Agglomerative Clustering

- **Heuristic**: If a set of RPs share similar profiles, the closed region of these RPs should not contain entities like walls and obstacles.
- Agglomerative clustering
  - Two small clusters are merged only if they're not separated by an entity.
- Better results



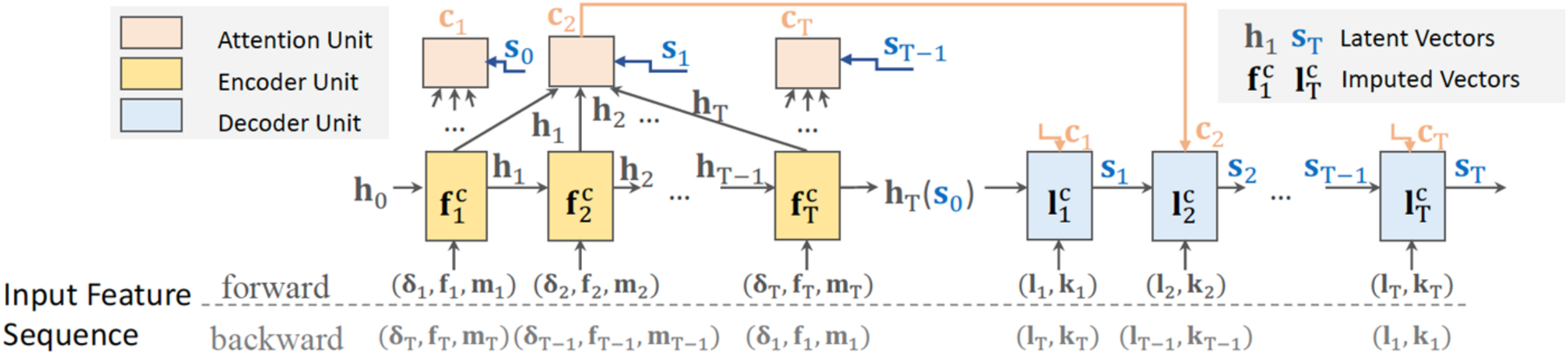
Imputer

- Task: impute MAR RSSI and missing RP jointly
- Intuitions
  - Radio map records on the same survey path are temporally correlated.
  - The fingerprint and RP in one record are also correlated.
- Ideas
  - Each survey path is a sequence with missing values.
  - We capture the correlations in a sequence-2-sequence data imputation.



Bi-directional Seq2seq Imputation Model

- BiSIM
  - Input: A sequence of T radio map records on a survey path
  - Output: A corresponding sequence of T imputed records



Fingerprint Input Feature

δ_i: time-lag vector
$\mathbf{f}_i$: fingerprint
$\mathbf{m}_i$: missing type mask vector

RP Input Feature

$\mathbf{l}_i$: RP
$\mathbf{k}_i$: mask vector

Experimental Settings (1)

- Our variants
  - **D-BiSIM**: K-means based differentiator + BiSIM
  - **T-BiSIM**: Topology-aware differentiator + BiSIM
- Baseline imputation methods
  - Missing RP only: Case Deletion (**CD**)/ Linear Interpolation (**LI**)/Semi-supervised Learning (**SL**)
  - Missing RSSI only: Multiple Imputation by Chained Equation (**MICE**)/Matrix Factorization (**MF**)
  - Bidirectional Recurrent Imputation for Time Series (**BRITS**)
  - Semi-Supervised Generative Adversarial Network (**SSGAN**)
- The imputed radio maps are used by three online location estimation algorithms
  - KNN
  - Weighted KNN (WKNN)
  - Random Forest (RF)

Experimental Settings (2)

- Two real indoor positioning datasets from Microsoft Research
  - Kaide Mall and Wanda Square

Venue	Kaide	Wanda
Floor Area (m^2)	3225.7	4458.5
RP density (per 100 m^2)	3.53	2.65
# of fingerprints	894	4104
# of RPs	114	118
# of APs (i.e., # of fingerprint dimensions)	671	929

- Overall performance metrics
  - Average Positioning Error (APE)
    - The distance between online estimated location and the true location
  - Data imputation time cost

Experimental Results

- Our BiSIMs result in the lowest APE (in meters)
  - Effective data imputation helps improve indoor positioning accuracy.

location estimation alg.	Kaide									Wanda								
	CD	LI	SL	MICE	MF	BRITS	SSGAN	D-BiSIM	T-BiSIM	CD	LI	SL	MICE	MF	BRITS	SSGAN	D-BiSIM	T-BiSIM
KNN	6.79	5.76	6.83	15.37	15.58	2.99	2.26	1.98	**1.78**	12.73	9.96	8.63	25.13	28.23	5.14	4.62	3.41	**2.43**
WKNN	6.64	5.76	7.10	15.37	15.65	3.07	2.23	1.96	**1.66**	12.52	9.95	8.45	27.91	28.35	4.78	3.47	3.27	**2.41**
RF	7.23	5.57	7.35	15.00	15.36	5.07	4.49	2.93	**2.70**	11.28	9.25	9.03	26.81	27.64	18.52	8.02	3.44	**3.10**

- Our BiSIMs time costs are not the longest

	LI	SL	MICE	MF	BRITS	SSGAN	D-BiSIM	T-BiSIM
Kaide	1.35	2.41	12.06	29.89	13.84	21.41	12.87	15.10
Wanda	2.38	5.50	22.64	67.02	23.13	33.43	22.74	25.43

Agenda

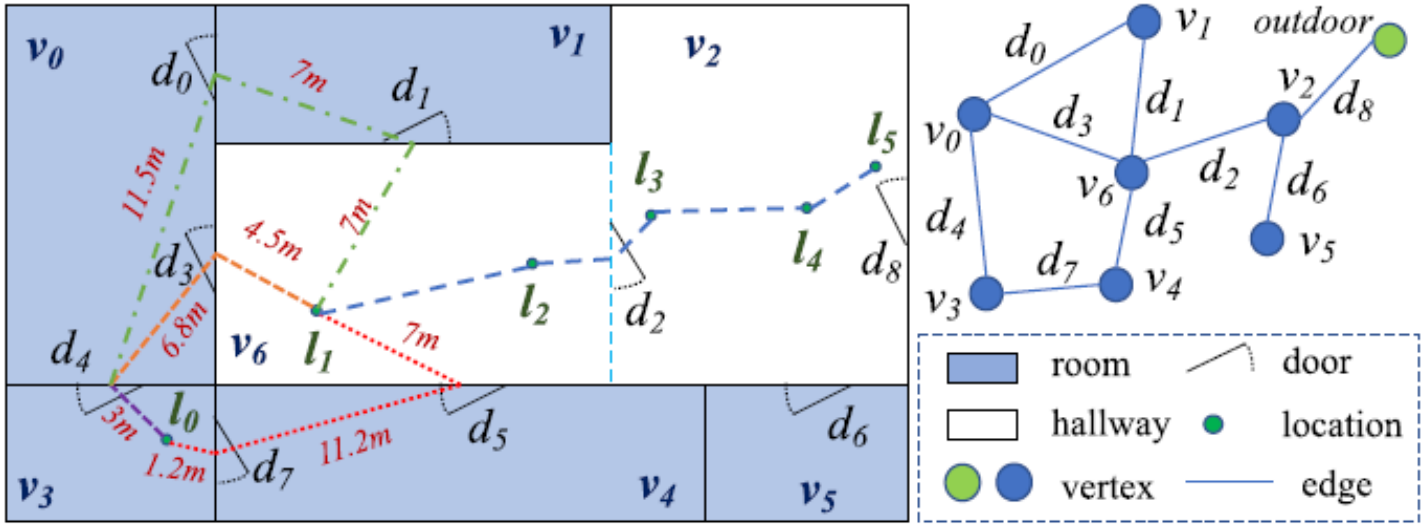
- Introduction
- Learned index for spatial data
- Representation learning of spatial data
- Machine learning for location-based services
  - Data imputation for indoor positioning
  - Indoor population modeling and monitoring
- Conclusion and future research

Motivation

- In large venues such as shopping malls and airports, knowledge on the indoor populations fuels applications
  - E.g., business analytics, venue management, and safety control.
- However, as we've seen, indoor positioning is not precise
  - Low sampling → *temporally* and *spatially* sparse indoor positioning data
  - It is hard to know the precise indoor populations

- We design techniques to use imprecise indoor positioning data for
  - Modeling populations in partitions of indoor space (*offline*)
  - Monitoring indoor populations continuously (*online*)

Preliminaries



- Indoor space
  - **Partition**: a unit of space, e.g., a room, or a hallway
  - Indoor Graph $G=(V, E)$: V for partitions and E for edges of adjacency
- Indoor positioning
  - Positioning record (o, l, t): Object o was seen at location l at time t.
  - Trajectory: $tr = \langle (l_1, t_1), \ldots (l_n, t_n) \rangle$, with $t_1 < t_2 < \ldots < t_n$.
- Indoor path
  - An interconnected door sequence between a source location l_s and a target location l_e. Formally, $\phi = \langle l_s, d_1, \ldots, d_m, l_e \rangle$

Preliminaries, cont.

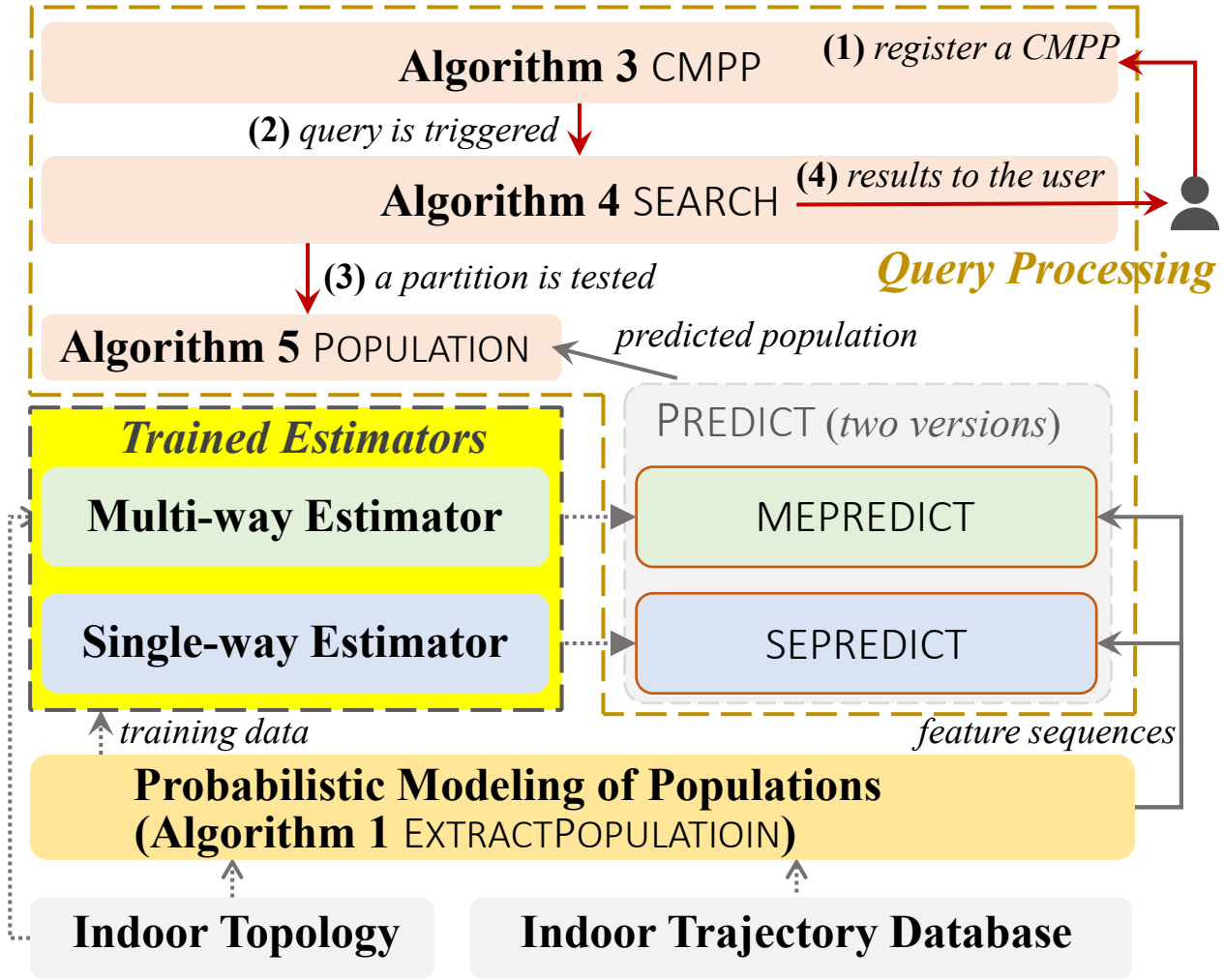
- Partition v 's **population** at time t: $P_{v,t}$
  - The number of moving objects in v at time t
- Populated Partition
  - Given a population threshold $\theta \in Z^+$ and a confidence threshold $\eta \in (0, 1)$, a partition v is a **populated partition** at time t, if the probability mass function (PMF) f of its population $P_{v,t}$ satisfies:
    - $f(P_{v,t} > \theta) \geq \eta$, i.e., $P_{v,t}$ exceeds θ with a probability at least η.
- Our research objectives
  - Modelling and monitoring of indoor population

Research Problems

- Problem 1: **Population Model**
  - Given a trajectory database $TR = \{tr_i = <(l_1, t_1), \ldots (l_n, t_n)>\}$ and a set of concerned partitions $V^* \subseteq V$, decide the *distribution* of the population $P_{v,t}$ of each partition $v \in V^*$ at a specified historical timestamp t within the scope of TR.
- Problem2: **Continuous Monitoring of Populated Partitions** (**CMPP**)
  - Given a distance range r, a population threshold θ, a confidence threshold η, an end time t_{end}, and a query time interval Δt, a CMPP query returns a list of populated partitions satisfying the thresholds θ and η within the range r of the current query location when
    1) the current time is no older than t_{end},
    2) the query location is changing, and
    3) the time since the most recent result update exceeds Δt.

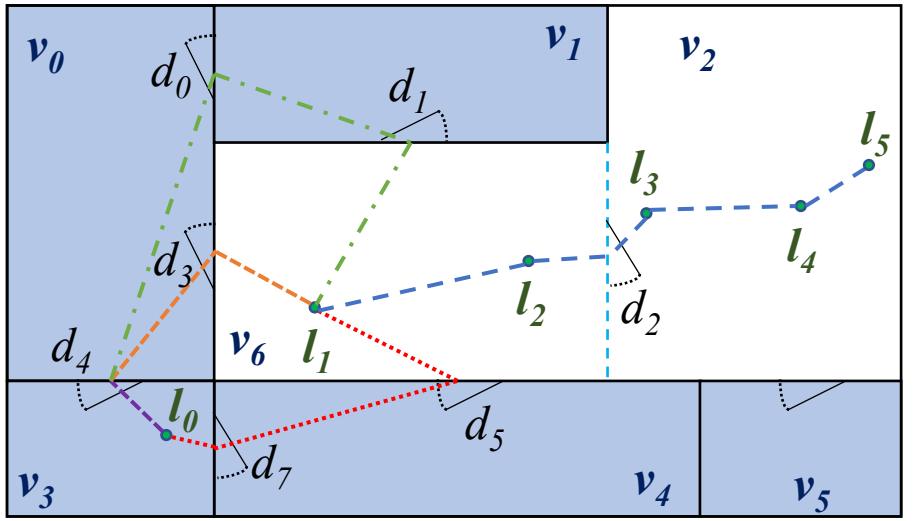
Overall Idea and Solution Framework

- Input
  - Indoor topology
  - Indoor trajectory database
- Main components
  - Probabilistic modeling of populations
    - From historical data
  - Partition population estimators
    - Multi-way and single-way
  - Population monitoring
    - Calling population predictors based on the estimators
    - Caching to improve performance



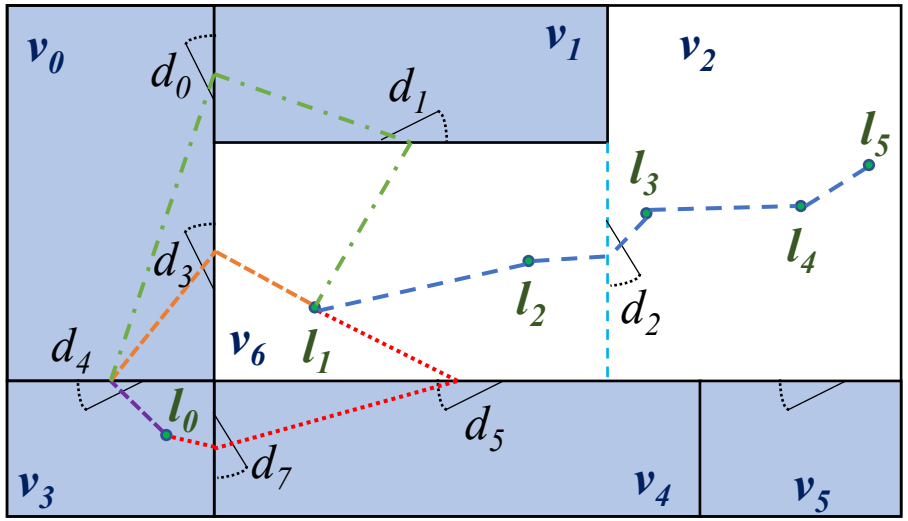
Probabilistic Modeling of Populations (1)

- Raw trajectory $tr = \langle (l_1, t_1), \ldots (l_n, t_n) \rangle$
  - Locations l_i and l_j may be in different partitions. Multiple paths may exist between them.
- The possible cross-door movement between two consecutive locations l_A and l_B
  - A random variable path $\phi = \langle l_A, d_1, \ldots, d_m, l_B \rangle$
  - ϕ follows a categorical distribution with a probability mass function $f(\phi = \phi_j) = Pr(\phi_j), j = 1, \ldots, J.$
    - J is the number of possible paths
    - $Pr(\phi_j)$ is the occurrence probability of the jth possible path ϕ_j
    - $\sum Pr(\phi_j) = 1$



Probabilistic Modeling of Populations (2)

- Suppose l_A (resp. l_B) is observed time t_A (resp. t_B)
- At a particular time t between t_A and t_B, object o's presence in a partition v is captured as a probability:
  - $Pr(o \mid v, t) = \sum Pr(\phi_j) \cdot Pr(v \mid \phi_j, t)$
    - $Pr(v \mid \phi_j, t)$ models the probability that path ϕ_j passes partition v at time t.
    - It is non-deterministic due to random movements.
    - We estimate it based on the Monte Carlo sampling over the possible time interval within which the object passes a door along ϕ_j.
  - Object o's presence in partition v at time t is a Bernoulli variable $\Omega_{o,v,t} \in \{0, 1\}$ with success probability $Pr(o \mid v, t)$.
    - The success probability is *non-identical* across all objects.



Probabilistic Modeling of Populations (3)

- Suppose O is the full set of moving objects in the space at time t.
- Population of a partition v at a time t: $P_{v,t} = \sum_{o \in O} \Omega_{o,v,t}$
  - $P_{v,t}$ is an integer p between 0 and $|O|$
  - $P_{v,t}$ has $(|O| + 1)$ possible categories, i.e., $p(0), \ldots, p(|O|)$.
  - As a sum of non-identically distributed Bernoulli variables of all moving objects, $P_{v,t}$ follows the Poisson Binomial distribution with the following PMF:
    - $f(P_{v,t} = p) = \sum_{C_p \in \mathbb{C}(p)} \prod_{o_i \in C_p} Pr(o_i \mid v, t) \prod_{o_j \in O \setminus C_p} (1 - Pr(o_j \mid v, t))$
      - $\mathbb{C}(p)$: Set of all possible combinations of the p objects in v at time t. $|\mathbb{C}(p)| = C(|O|, p)$
      - $C_p \in \mathbb{C}(p)$ refers to one possible combination
  - A Poisson Binomial distribution can be approximated as a Normal distribution. Population can be approximated as a Normal distribution $P_{v,t} \sim \mathcal{N}(\mu, \sigma^2)$
    - Mean: $\mu = \sum_{o \in O} Pr(o \mid v, t)$
    - Variance: $\sigma^2 = \sum_{o \in O} Pr(o \mid v, t) \cdot (1 - Pr(o \mid v, t))$

Probabilistic Modeling of Populations (4)

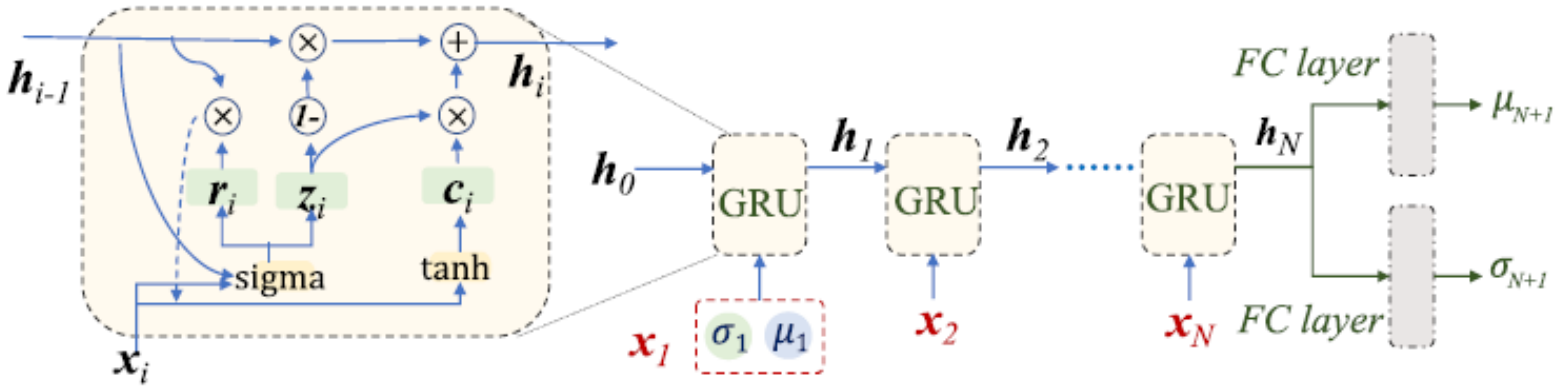
- To estimate (μ, σ^2) for $P_{v,t} \sim \mathcal{N}(\mu, \sigma^2)$, we need $Pr(o \mid v, t)$ for individual objects
  - Recall $Pr(o \mid v, t) = \sum Pr(\phi_j) \cdot \Pr(v \mid \phi_j, t)$
- Thus, we must have
  - $Pr(\phi_j)$, i.e., path ϕ_j's occurrence probability
  - $Pr(v \mid \phi_j, t)$, i.e., the probability to pass partition v via path ϕ_j at time t
- We generate these probabilities by using the historical data
  - For each consecutive reported locations (l_A, l_B)
    - For each possible path ϕ between l_A and l_B
      - We use Monte Carlo sampling to find a possible partition that ϕ passes at time t
  - For each partition v, we calculate its (μ, σ^2) at time t

Population Estimators

- From the historical data, we obtain the historical means and variances.
- Based on the historical parameter values, how to estimate the current (or future) means and variances?
  - A single-way estimator predicts the population for only one partition
  - A multi-way estimator can predict the populations for all partitions in one pass

Single-Way Estimator (SE)

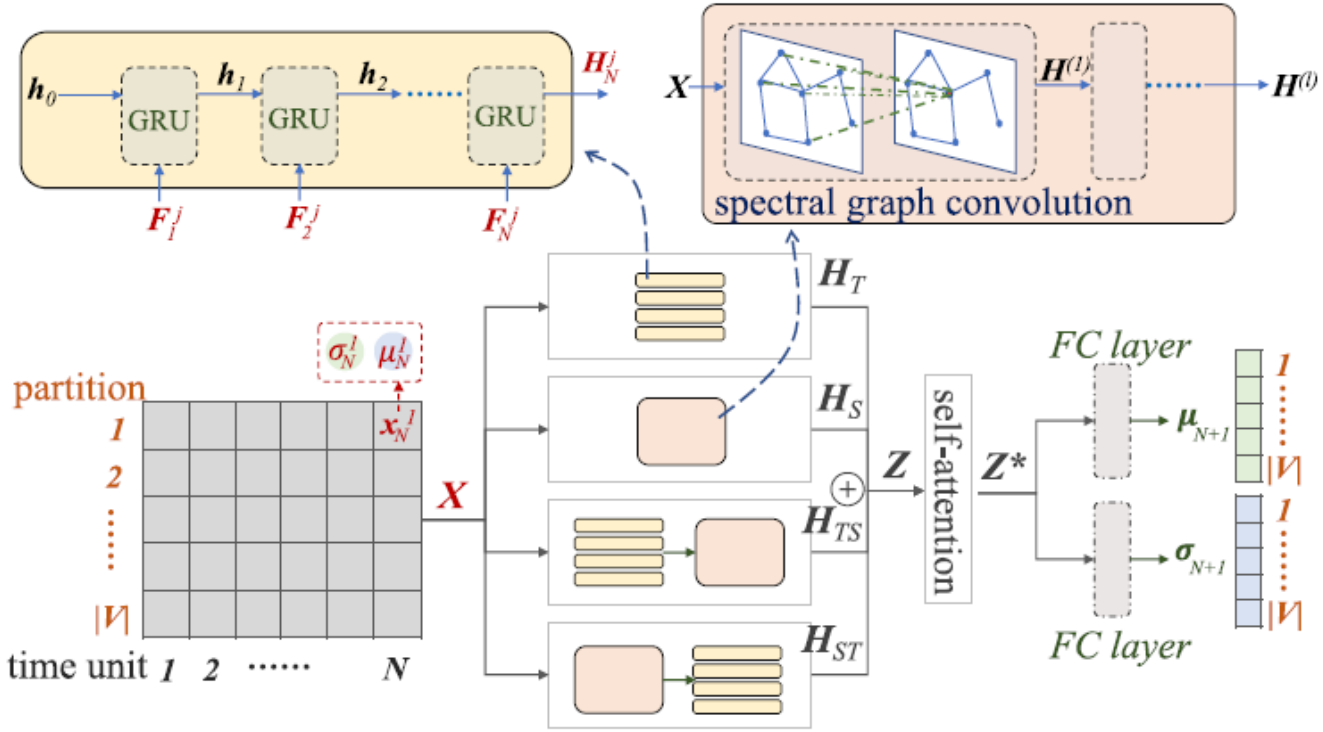
- Model architecture (for a partition v at the current timestamp t)
  - Gated Recurrent Unit (GRU) extended with two parallel fully connected (FC) layers for outputs.
- Input
  - A sequence of v's N most recent historical populations, i.e., feature sequence
  - $x_i = [\mu_{v,t-(N-i+1)\delta}, \sigma_{v,t-(N-i+1)\delta}]$
- Outputs
  - The mean $\mu_{v,t}$ and standard deviation $\sigma_{v,t}$ of v's current population at t.



- **Loss function**
  - MSE for mean + MSE for stand deviation
- **Training of SE**
  - On historical populations in a single partition
  - *However*, a partition's current population is related to its adjacent partitions.

Multi-Way Estimator (ME)

- Input
  - All partitions' population information at the N recent historical timestamps
- Output
  - Their current populations
- Internal
  - Four parallel units
    - $\boldsymbol{H}_T$: Temporal dependency
    - $\boldsymbol{H}_S$: Spatial dependency
    - $\boldsymbol{H}_{TS}$: Temporal-spatial dependency
    - $\boldsymbol{H}_{ST}$: Spatial-temporal dependency
    - GRU for temporal, graph for spatial
  - Self attention
  - FC layers

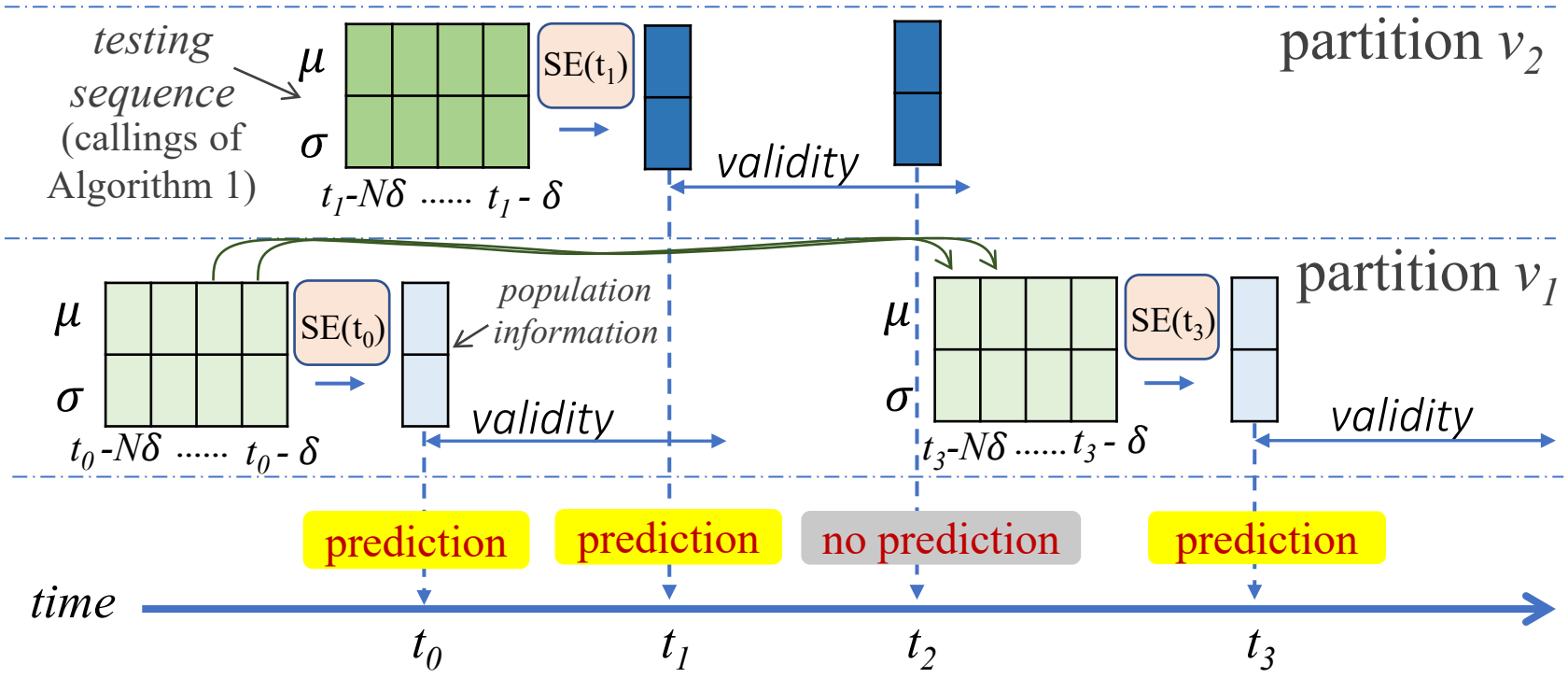


Estimator-Based Population Monitoring

- A query instance CMPP(r, θ, η, t_{end}, Δt)
  - Initialization: Get the current query location l_q and its host partition v
  - Starting from the current partition v, expand to each reachable partition within the distance r
    - Get the current partition v's population $P_{v,t}$ via an estimator (SE or ME)
    - Include v into the result if $f(P_{v,t} > \theta) \geq \eta$.
- Caching mechanism for all partitions encountered in a query
  - Hashtable $\mathcal{H}$: $V \rightarrow$ (*prob*, *time*)
    - *prob*: The last updated probability $f(P_{v,t} > \theta)$
    - *time*: The corresponding update time
  - The cached probability is valid if $t_q - \mathcal{H}[v].time \leq$ validity

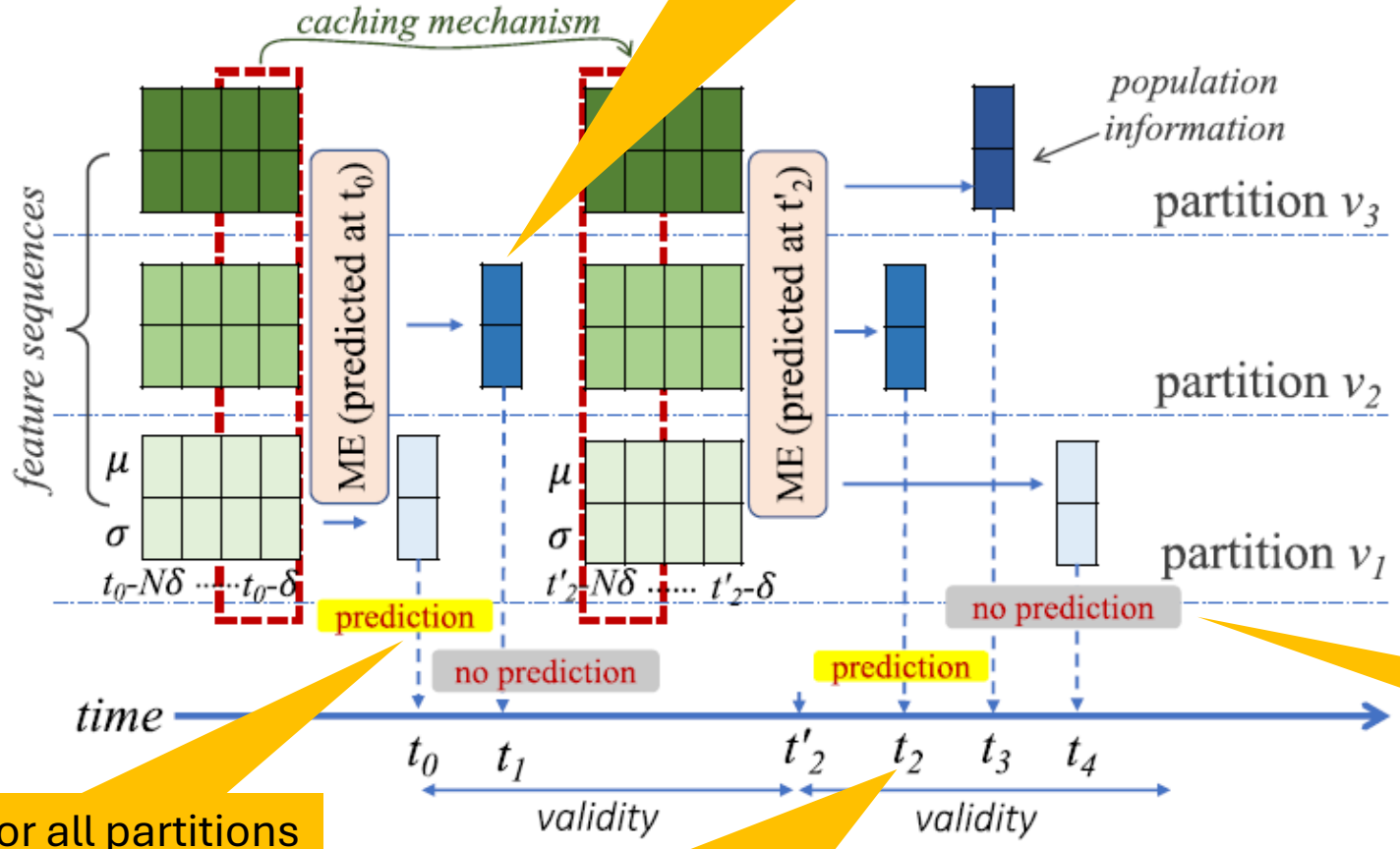
CMPP Processing with SE

- Only encountered partitions are cached.



CMPP Processing with ME

- All partitions are cached.



No prediction is needed within the cache validity.

Prediction for all partitions

Prediction for all partitions

No prediction is needed within the cache validity.

Experiments

- Real datasets from two buildings
  - January 2018

Data Set	BLD-1	BLD-2
# of floors	7	6
# of partitions	1050	900
# of doors	1613	1554
# of positioning records per day	91.1k	256k
# of trajectories per day	1,376	2,119

- Data split
  - 22 days (70%) for training, next 3 days (10%) for validation, last 6 days (20%) for test
- Number of Monte Carlo sampling: 200

Experimental Results: Estimators

- Performance metric: Kullback–Leibler (KL) divergence
- δ: the time window size between two consecutive GRUs.
- Baselines:
  - ARIMA: Auto-Regressive Integrated Moving Average
  - SVR: Supported Vector Regression
  - TGCN, STGCN, and ASTGNN: Graph neural networks

δ (min.)	BLD-1							BLD-2						
	ARIMA	SVR	TGCN	STGCN	ASTGNN	SE	ME	ARIMA	SVR	TGCN	STGCN	ASTGNN	SE	ME
1	11.28	17.36	8.35	2.85	3.71	1.39	**0.81**	18.01	22.53	9.56	2.56	3.75	1.85	**0.95**
5	13.39	18.54	9.64	4.83	7.12	2.11	**0.93**	23.58	30.86	18.25	6.67	12.39	2.76	**1.20**
10	23.34	20.45	15.24	7.92	17.96	3.23	**1.24**	43.46	35.57	32.66	12.67	27.58	3.81	**1.84**

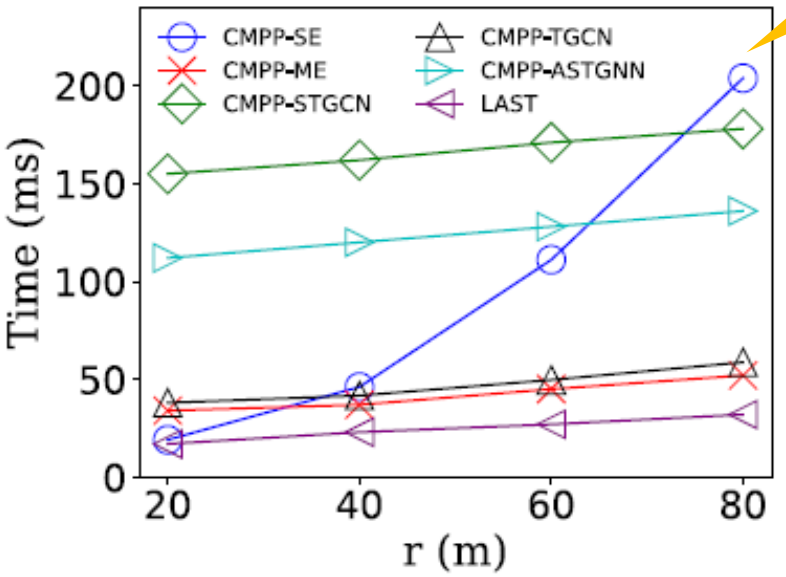
Experimental Results: CMPP Processing (1)

- Query settings

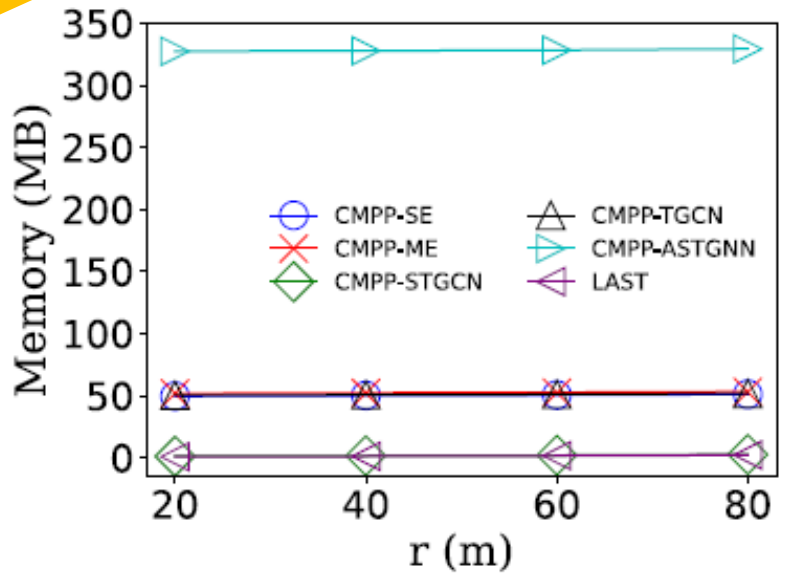
Parameter	Value
r (meter)	20, 40, **60**, 80
θ	**2**, 4, 6, 8
`Validity` (second)	60, **120**, 180, 240

- Baseline LAST
  - It uses the *last* observed object locations directly
  - Low cost but ineffective

CMPP-SE benefits less from caching when more partitions are involved



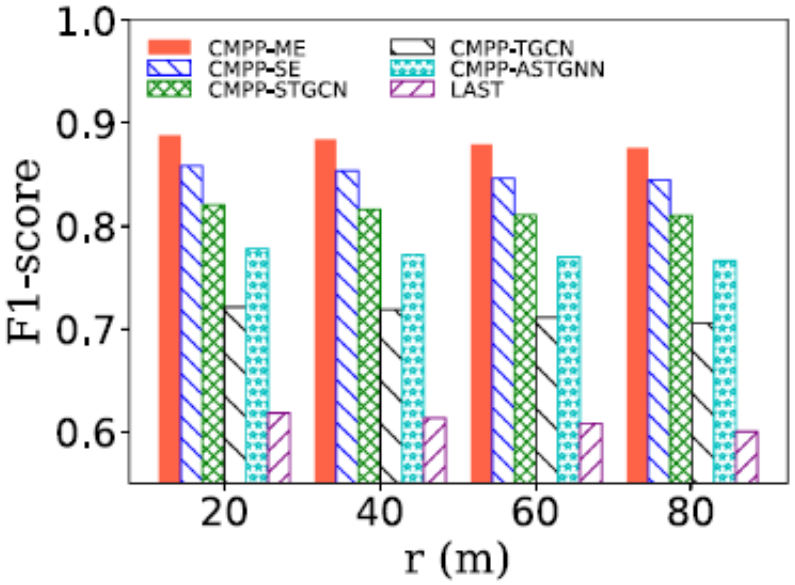
(a) Time vs r



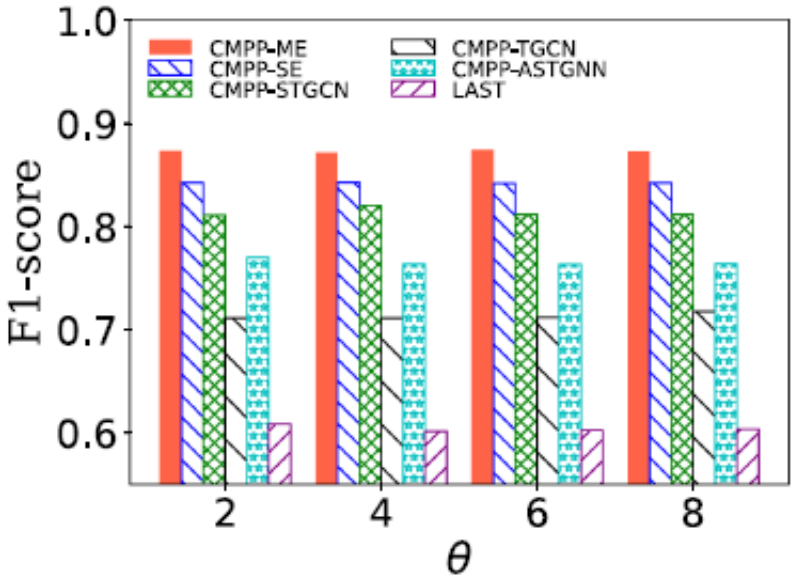
(b) Memory vs r

Experimental Results: CMPP Processing (2)

- CMPP-ME achieves the highest F1 score.
- It needs slightly more memory for the cache, which nevertheless pays off.
- Its time cost is also acceptable.



(c) F1-score vs r



(d) F1-score vs θ

Agenda

- Introduction
- Learned index for spatial data
- Representation learning of spatial data
- Machine learning for location-based services
- Conclusion and future research

Conclusion

- ML can effectively improve spatial data management
  - Learned spatial indexes (smaller but faster)
    - Simplistic design (Z-order + RMI) doesn’t work well
    - A more sophisticated design (LISA) works much better
  - Analytics (effectiveness of representation learning)
    - Trajectory data is an important part of spatial data
    - Unified representation learning of points, polylines and polygons
- ML can help location-based services as well
  - Improving (indoor) positioning accuracy
  - Enabling effective and efficient indoor population modeling and monitoring

Future Research Directions

- Learned spatial index
  - For polylines, polygons, or trajectories
  - Unified learned index for heterogenous spatial data
  - Accommodating data updates more effectively
- Representation learning of spatial data
  - Representation learning of multimodal trajectories
    - Urban mobility with driving, public transportation, bicycling, walking, etc.
    - Trajectories from land, sea and air
  - Alternative encoding approaches
- Other research topics
  - What (else) is needed for AI to achieve human-like spatial capabilities?
    - Without solid spatial capabilities, AGI cannot be true.

An example

- Current LLMs have limited spatial capabilities.
- Effective representation learning of spatial objects is a cornerstone for spatial capabilities of LLMs.
- And for GeoAI
  - Geospatial artificial intelligence: application of AI to geospatial data

Draw a cartoon picture that shows a flight from Brussels in Europe to Chicago in US, and this flight is over the North Atlantic Ocean.

Image created

Inspired by Song Wu, ULB

Acknowledgments

- Students
  - Haixin Wang (visiting research student from HKBU)
  - Pengfei Li, Xiao Li, Jialiang Li (PhDs)
- Postdocs
  - Huan Li, Harry Kai-Ho Chan, Tiantian Liu
- All coauthors
- Sponsors

AALBORG
UNIVERSITET

References

Learned spatial indexes

1. Haixin Wang, Xiaoyi Fu, Jianliang Xu, Hua Lu: Learned Index for Spatial Queries. MDM 2019: 569-574
2. Pengfei Li, Hua Lu, Qian Zheng, Long Yang, Gang Pan: LISA: A Learned Index Structure for Spatial Data. SIGMOD Conference 2020: 2119-2133

Representation learning

3. Jialiang Li, Tiantian Liu, Hua Lu: CLEAR: Ranked Multi-Positive Contrastive Representation Learning for Robust Trajectory Similarity Computation. MDM 2024: 21-30 (**Best paper runner-up award**)
4. Maria Despoina Siampou, Jialiang Li, John Krumm, Cyrus Shahabi, Hua Lu: Poly2Vec: Polymorphic Fourier-Based Encoding of Geospatial Objects for GeoAI Applications. ICML 2025

Machine learning for LBS

5. Xiao Li, Huan Li, Harry Kai-Ho Chan, Hua Lu, Christian S. Jensen: Data Imputation for Sparse Radio Maps in Indoor Positioning. ICDE 2023: 2235-2248
6. Xiao Li, Huan Li, Hua Lu, Christian S. Jensen: Modeling and Monitoring of Indoor Populations Using Sparse Positioning Data. IEEE Trans. Knowl. Data Eng. 37(2): 794-809 (2025)

Location inference in SoMe

7. Pengfei Li, Hua Lu, Qian Zheng, Shijian Li, Gang Pan: HisRect: Features from Historical Visits and Recent Tweet for Co-Location Judgement. IEEE Trans. Knowl. Data Eng. 33(3): 1005-1018 (2021)
8. Pengfei Li, Hua Lu, Nattiya Kanhabua, Sha Zhao, Gang Pan: Location Inference for Non-Geotagged Tweets in User Timelines. IEEE Trans. Knowl. Data Eng. 31(6): 1150-1165 (2019)