



**KOÇ
UNIVERSITY**

Grid-Based Decompositions for Spatial Data under Local Differential Privacy

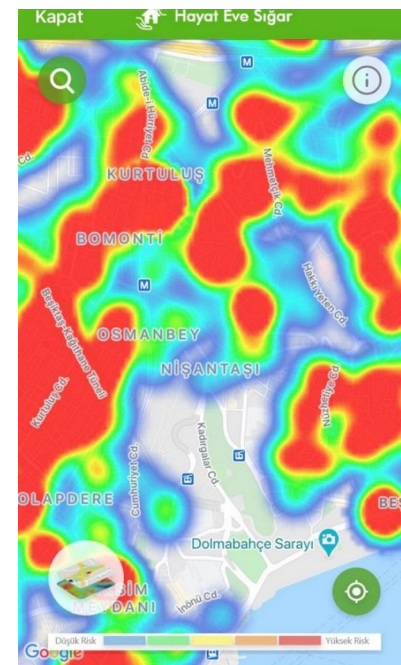
**Berkay Kemal Balioglu, Alireza Khodaie,
Ameer Taweel, M. Emre Gürsoy**

Department of Computer Engineering
Koç University, Istanbul, Turkey



Spatial Data and Privacy

- Large volumes of spatial data are nowadays available for collection and analysis
 - Smartphones
 - Connected cars
 - Social networks
 - Location-based services



- However, spatial data is **privacy-sensitive**

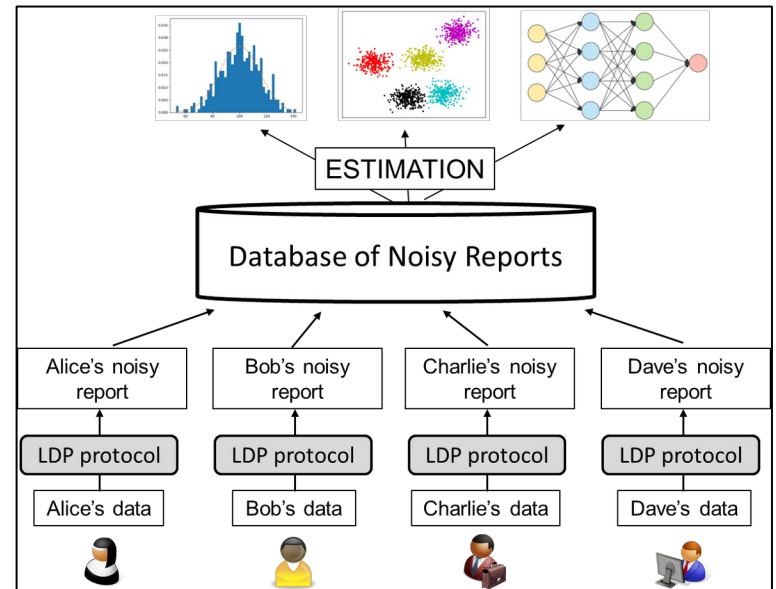
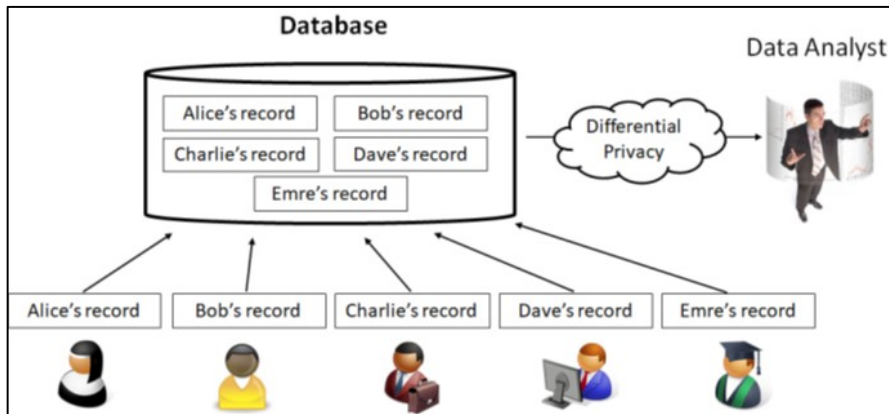


How location tracking is raising the stakes on privacy protection



DP and LDP

- In recent years, **Differential Privacy (DP)** and **Local Differential Privacy (LDP)** have become two popular standards for privacy-preserving data analysis

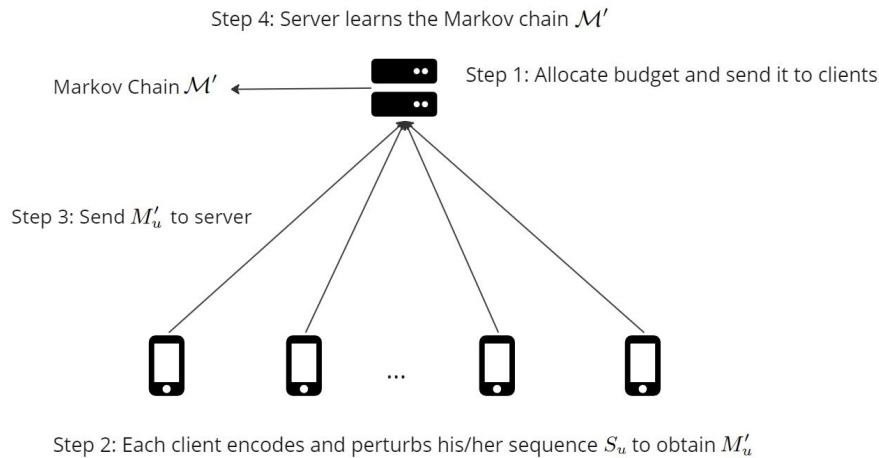


- Due to its popularity, there is rising interest and recent work towards applying **LDP** to **spatial data analysis**



Grids in LDP

- Grid-based decompositions have been a common building block when applying DP and LDP to spatial data

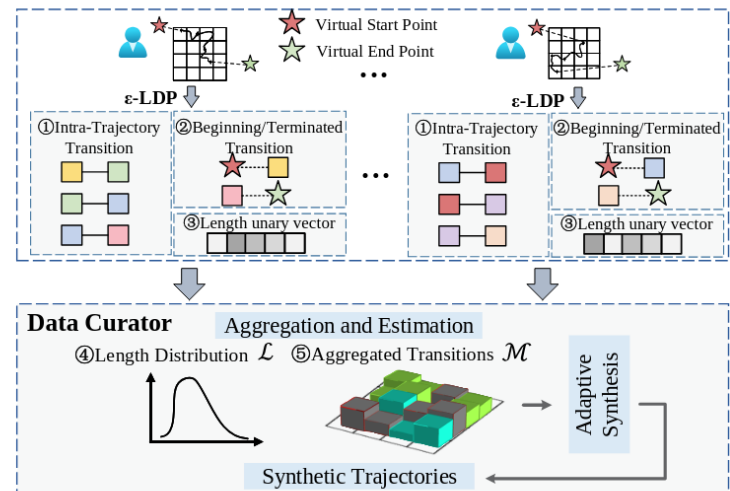


Guner et al. (ESORICS 2023) Learning Markov Chain Models from Sequential Data under LDP

Uniform & adaptive grids first proposed by Qardaji et al. (ICDE 2013) for DP.

Later used in many other works:

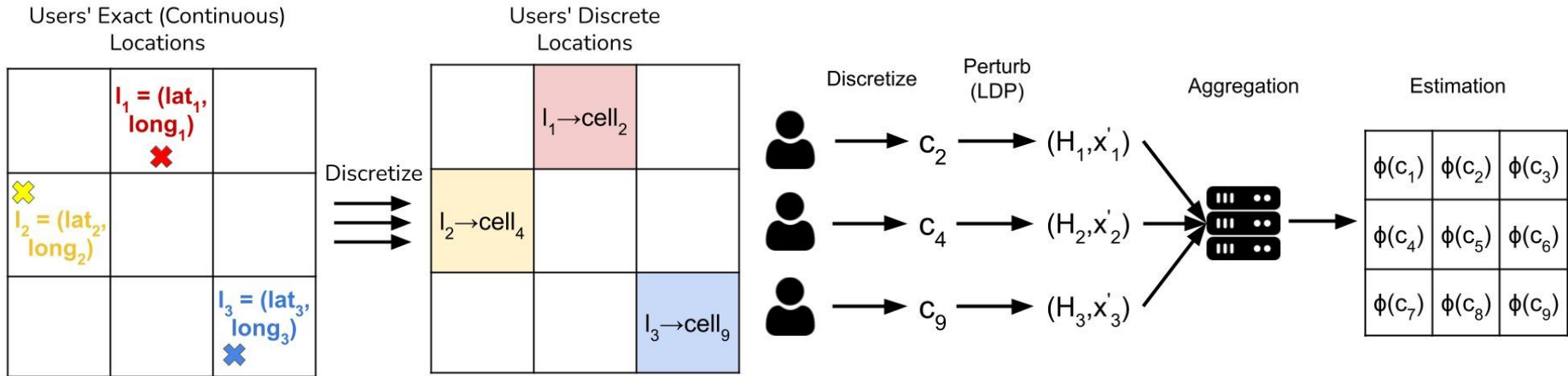
- He et al. (VLDB 2015)
- Gursoy et al. (CCS 2018)
- Wang et al. (Usenix Security 2023)
- ...



Du et al. (VLDB 2023) LDPTrace: Locally Differentially Private Trajectory Synthesis



Uniform Grid



Algorithm 1 Collecting location data with LDP using a grid

- 1: **Input:** Users \mathcal{U} , grid \mathcal{G} , privacy budget ϵ
 - 2: **Output:** Densities of each cell in \mathcal{G}
 - 3:
 - 4: \triangleright **User-side discretization and perturbation**
 - 5: **for** each user $u_i \in \mathcal{U}$ **do**
 - 6: **for** each cell $C_j \in \mathcal{G}$ **do**
 - 7: **if** l_i falls inside C_j **then**
 - 8: Set user's true cell as: $C_i \leftarrow C_j$
 - 9: **break**
 - 10: Execute the user-side OLH protocol with true value = C_i , domain $\mathcal{D} = \mathcal{G}$, and privacy budget = ϵ
 - 11: Send the resulting tuple $\langle H_u, x'_u \rangle$ to the server
 - 12:
 - 13: \triangleright **Server-side estimation**
 - 14: Server receives $\langle H_u, x'_u \rangle$ from all $u_i \in \mathcal{U}$
 - 15: **for** each cell $C_j \in \mathcal{G}$ **do**
 - 16: Compute $\text{Sup}(C_j)$ as the number of tuples for which $x'_u = H_u(C_j)$
 - 17: Compute estimate $\Phi(C_j)$ using Equation 3
 - 18: **return** $\Phi(C_1), \Phi(C_2), \dots$ for all $C_j \in \mathcal{G}$
-

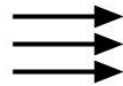
- Each user discretizes his/her location into a grid cell
- Perturb and send using an LDP protocol, e.g., OLH protocol
- Server receives perturbed responses from all users
- Perform estimation to recover density estimates of each cell



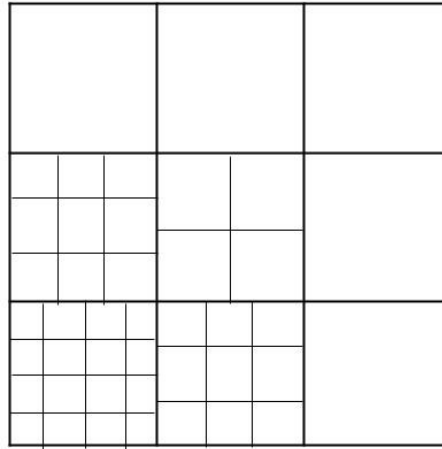
Adaptive Grid

First Phase Estimated Cell Densities

$\phi(c_1)=$ 49	$\phi(c_2)=$ 43	$\phi(c_3)=$ 18
$\phi(c_4)=$ 1017	$\phi(c_5)=$ 774	$\phi(c_6)=$ 21
$\phi(c_7)=$ 1256	$\phi(c_8)=$ 998	$\phi(c_9)=$ 69

Adapt


Second Phase Adaptive Grid



$$g_1 = \sqrt{2\alpha \cdot (e^\varepsilon - 1) \cdot \sqrt{\frac{|\mathcal{U}|}{e^\varepsilon}}}$$

$$g_2^k = \sqrt{2\alpha \cdot \Phi(C_k) \cdot (e^\varepsilon - 1) \cdot \sqrt{\frac{(1 - \sigma) \cdot |\mathcal{U}|}{e^\varepsilon}}}$$

Algorithm 2 Algorithmic summary of the PrivAG approach

- 1: **Input:** Users \mathcal{U} , parameters α and σ , privacy budget ε
 - 2: **Output:** Densities of each cell in adaptive grid \mathcal{G}_{ag}
 - 3:
 - 4: **▷ First phase of PrivAG**
 - 5: Server computes g_1 according to Equation 4
 - 6: Server divides \mathcal{U} into two groups \mathcal{U}_1 and \mathcal{U}_2 such that $|\mathcal{U}_1| = \sigma \times |\mathcal{U}|$
 - 7: Server lays $g_1 \times g_1$ uniform grid \mathcal{G}_1 on Ω
 - 8: Call Algorithm 1 with \mathcal{U}_1 , \mathcal{G}_1 and ε to obtain $\Phi(C_1), \Phi(C_2), \dots$ for all $C_k \in \mathcal{G}_1$
 - 9:
 - 10: **▷ Second phase of PrivAG**
 - 11: **for** each cell $C_k \in \mathcal{G}_1$ **do**
 - 12: Compute g_2^k according to Equation 5
 - 13: Divide C_k into $g_2^k \times g_2^k$ cells of equal size
 - 14: Let \mathcal{G}_{ag} denote the resulting grid after the above divisions
 - 15: Call Algorithm 1 with \mathcal{U}_2 , \mathcal{G}_{ag} and ε to obtain $\Phi(C_1), \Phi(C_2), \dots$ for all cells in \mathcal{G}_{ag}
-

- In the first phase, collect data using a uniform grid to estimate cells' densities
- In the second phase, adapt the grid according to densities obtained in the first phase



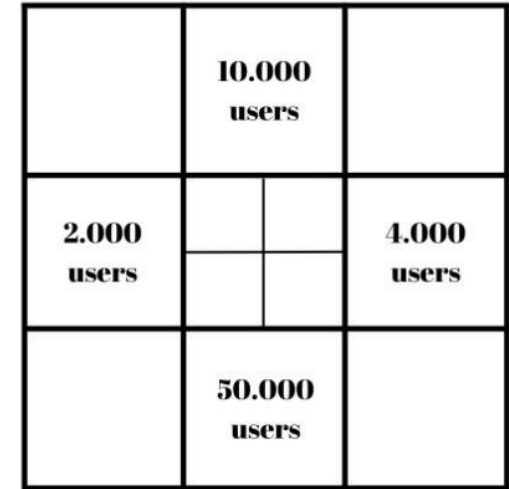
Advanced Adaptive Grid (AAG)

Shortcomings of Yang et al. (2022)'s adaptive grid **PrivAG**:

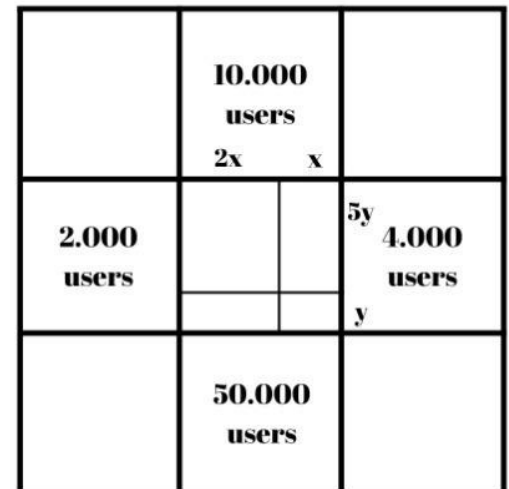
- Number of cells is too similar to that of the initial grid
- When dividing, does not take neighbouring cells into account

Dataset	Grid	$\epsilon = 0.5$	$\epsilon = 1$	$\epsilon = 3$	$\epsilon = 5$
Gowalla	Initial \mathcal{G}_1	36	81	324	900
	PrivAG	42	96	440	1358
	AAG	276	567	2451	8745
Porto	Initial \mathcal{G}_1	25	49	225	625
	PrivAG	31	63	256	745
	AAG	188	390	1567	4503
Foursquare	Initial \mathcal{G}_1	16	36	121	361
	PrivAG	19	45	162	455
	AAG	107	235	957	2794

PrivAG



AAG





Advanced Adaptive Grid (AAG)

Key components of our **AAG**:

- Division of the current cell is done by also taking into account neighboring cells' densities
- Use of different parameters in g_1 and g_2^k
- Handling edge and corner cells

$$vsplit = \frac{\Phi(C_k^B)}{\Phi(C_k^U) + \Phi(C_k^B)} \times (\text{height of } C_k)$$

$$hsplit = \frac{\Phi(C_k^R)}{\Phi(C_k^L) + \Phi(C_k^R)} \times (\text{width of } C_k)$$

AAG

	10.000 users 2x x	
2.000 users		5y 4.000 users
		y
	50.000 users	

Algorithm 3 Algorithmic summary of the AAG approach

- 1: **Input:** Users \mathcal{U} , parameters α and σ , privacy budget ε
 - 2: **Output:** Densities of each cell in adaptive grid \mathcal{G}_{aag}
 - 3:
 - 4: \triangleright **First phase of AAG**
 - 5: The first phase of AAG is the same as PrivAG
 - 6:
 - 7: \triangleright **Second phase of AAG**
 - 8: **for** each cell $C_k \in \mathcal{G}_1$ **do**
 - 9: Compute g_2^k according to Equation [5](#)
 - 10: Compute $hsplit$ for C_k according to Equation [6](#)
 - 11: Compute $vsplit$ for C_k according to Equation [7](#)
 - 12: Divide C_k into four subcells using $hsplit$ and $vsplit$
 - 13: **if** $g_2^k > 2$ **then**
 - 14: Uniformly divide each subcell into $\frac{g_2^k - 1}{2}$ pieces horizontally and vertically
 - 15: Let \mathcal{G}_{aag} denote the resulting grid after the above divisions
 - 16: Call Algorithm [1](#) with \mathcal{U}_2 , \mathcal{G}_{aag} and ε to obtain $\Phi(C_1)$, $\Phi(C_2)$, .. for all cells in \mathcal{G}_{aag}
-



Experiment Setup

- All implementations were done in Python
- Three datasets: Porto, Gowalla, Foursquare
 - **Porto:** From ECML-PKDD Taxi Service Prediction competition, used locations which are inside the city of Porto
 - **Gowalla:** Location based social networking site, used check-ins made in the United States
 - **Foursquare:** Check-ins of social media users in Tokyo

● Error metric:

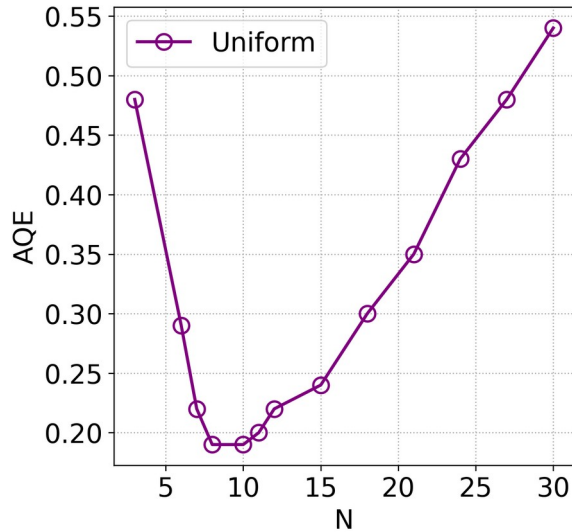
$$AQE = \frac{1}{\gamma} * \sum_{i=1}^{\gamma} \frac{|ans_{q_i} - ans'_{q_i}|}{\max\{ans_{q_i}, b\}}$$

$\gamma = 500$, number of random queries

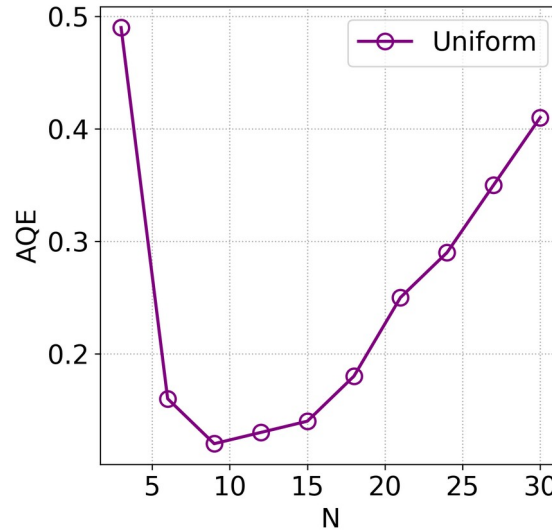
$b = 2\% \times (\text{number of users})$, to mitigate dominating effect of extremely selective queries



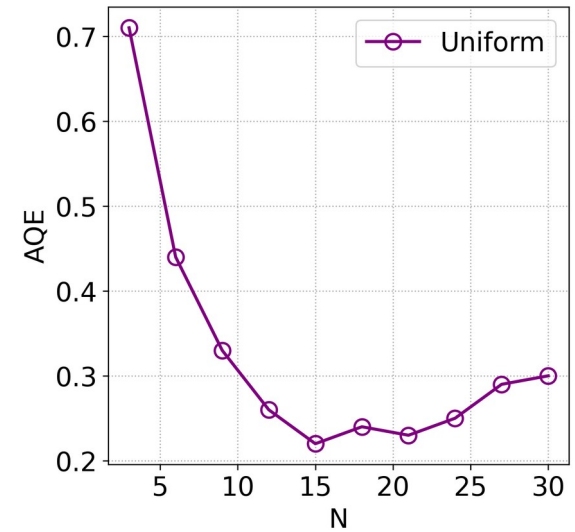
Experiments - Privacy Budget



Foursquare dataset



Porto dataset



Gowalla dataset

Dataset	Method	$\epsilon = 0.5$	$\epsilon = 1$	$\epsilon = 3$	$\epsilon = 5$
Gowalla	UG	0.0070	0.0066	0.0064	0.0056
	PrivAG	0.0075	0.0077	0.0072	0.0062
	AAG	0.0047	0.0051	0.0049	0.0041
Porto	UG	0.0048	0.0045	0.0045	0.0045
	PrivAG	0.0058	0.0056	0.0059	0.0060
	AAG	0.0048	0.0045	0.0049	0.0049
Foursquare	UG	0.0055	0.0054	0.0051	0.0048
	PrivAG	0.0060	0.0062	0.0061	0.0069
	AAG	0.0044	0.0043	0.0040	0.0047

Query size
(ρ) is fixed
to 0.01%



Experiments - Varying ρ

Dataset	Method	$\rho = 0.005\%$	$\rho = 0.01\%$	$\rho = 0.05\%$	$\rho = 0.1\%$	$\rho = 0.5\%$
Gowalla	UG	0.0034	0.0067	0.0279	0.0485	0.120
	PrivAG	0.0039	0.0077	0.0374	0.0728	0.305
	AAG	0.0023	0.0051	0.0236	0.0460	0.185
Porto	UG	0.0028	0.0045	0.0180	0.0321	0.082
	PrivAG	0.0034	0.0056	0.0283	0.0540	0.205
	AAG	0.0025	0.0045	0.0247	0.0501	0.195
Foursquare	UG	0.0032	0.0054	0.0243	0.0416	0.126
	PrivAG	0.0036	0.0062	0.0291	0.0547	0.203
	AAG	0.0025	0.0043	0.0234	0.0450	0.177

Dataset	Method	$\rho = 2\%$	$\rho = 4\%$	$\rho = 6\%$	$\rho = 8\%$	$\rho = 10\%$
Gowalla	UG	0.20	0.21	0.24	0.22	0.21
	PrivAG	0.83	1.15	1.32	1.29	1.25
	AAG	0.52	0.71	0.73	0.73	0.72
Porto	UG	0.12	0.12	0.11	0.12	0.09
	PrivAG	0.54	0.76	0.85	0.91	0.91
	AAG	0.38	0.49	0.60	0.61	0.65
Foursquare	UG	0.17	0.20	0.19	0.22	0.21
	PrivAG	0.48	0.63	0.68	0.70	0.71
	AAG	0.40	0.56	0.64	0.70	0.71

In both tables, ε is fixed to 1



Conclusions

- We studied three grid-based decomposition approaches under LDP: Uniform Grid (UG), PrivAG, and AAG
- Our proposed AAG approach advances the state-of-the-art adaptive grid approach (PrivAG) by performing cell divisions by taking into account the neighboring cells' densities
- We experimentally compared UG, PrivAG, and AAG using three datasets and multiple parameter values (ϵ and ρ)
 - AAG always beats PrivAG, and it also beats UG when ρ is small
 - When ρ is large, UG with a near-optimal choice of grid size becomes better than AAG
- **Future Work:**
 - We will investigate methods to improve PrivAG and AAG's utility especially in high ϵ regimes



Thank you!



Berkay Kemal Balioglu
bbalioglu23@ku.edu.tr



Alireza Khodaie
akhodaie22@ku.edu.tr



Ameer Taweel
ataweel20@ku.edu.tr



M. Emre Gürsoy
emregursoy@ku.edu.tr
www.memregursoy.com