Association for Computing Machinery

August 3-7, 2025 KDD2025 TORONTO

FedVS: Towards Federated Vector Similarity Search with Filters

Zeheng Fan, Yuxiang Zeng, Zhuanglin Zheng, Binhan Yang, Yongxin Tong
Beihang University

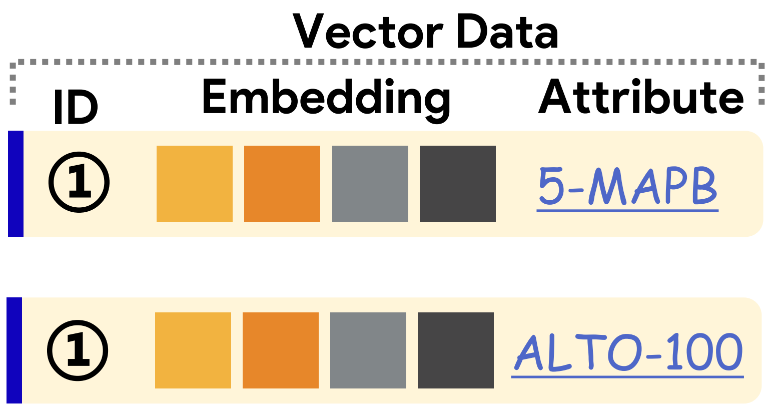
北京航空航天大學 BEIHANG UNIVERSITY

Outline

- Background
- Problem Statement
- Methodology
- Experiment
- Conclusion

Background: vector data

- Vector data : a hybrid data type that integrates both high-dimensional embeddings and structured attributes
  - **high-dimensional embeddings**: captures the intrinsic features of entity with deep models like BERT [1]
  - **structured attributes**: provide additional context or metadata associated with the entity

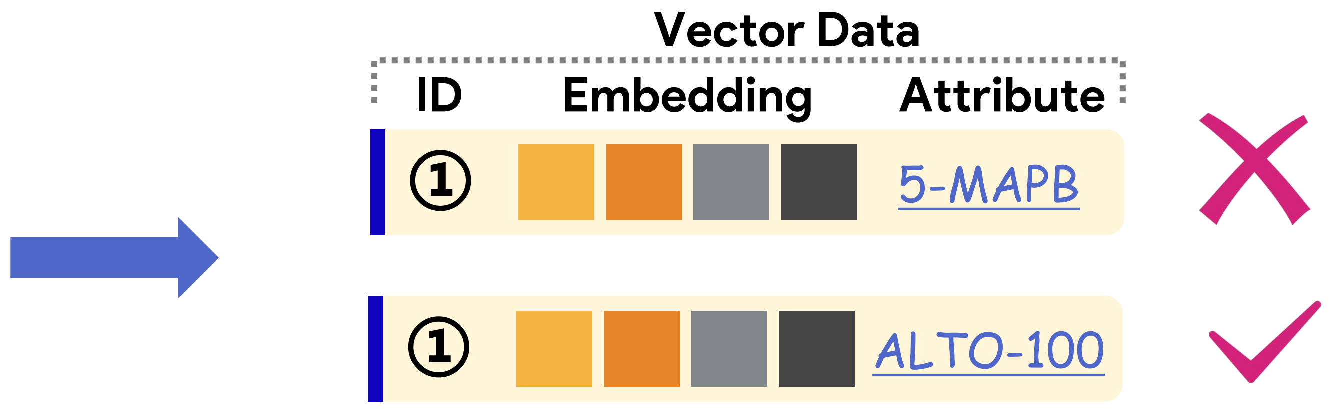


[1] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[C]//In NAACL-HLT, 2019: 4171-4186.

Background: vector similarity search

- Vector similarity search is composed of (q, k, P):
  - their attributes must match the attribrite filter P.
  - they are the � nearest neighbors (kNNs) to the query vector within the set of filtered data objects

Query vector : [0, 0, 0, …], $\boldsymbol{k}$ **:** 3
Attribute filter : “Drug == ALTO-100”

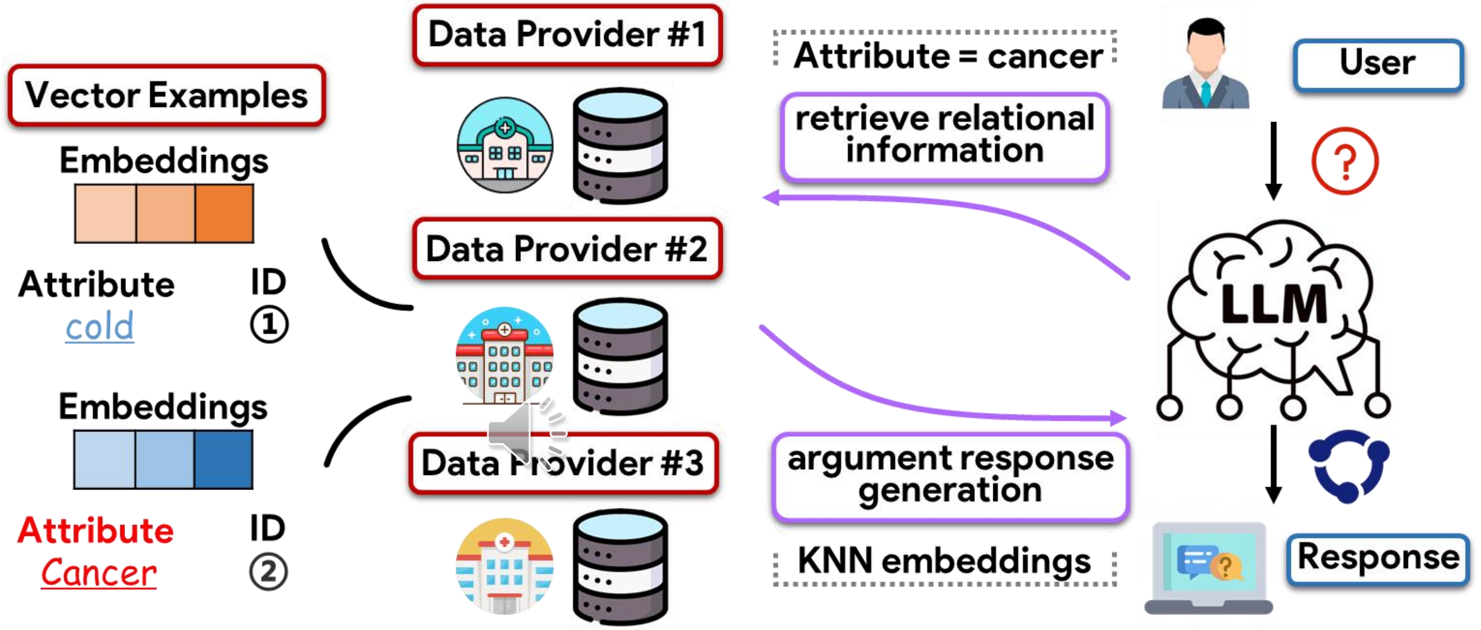


Background: federated vector similarity search

- Search over single-sourced dataset
  - Both industry and academia have developed efficient solutions
  - However, these solutions only focus on single-sourced vector data
- Search on federated dataset
  - Searching on **Multi-sourced** autonomous vector datasets **without exposing data privacy**
  - Data providers collerabrately provide a vector retrieval service over their union dataset

Background: federated vector similarity search

- Search on federated dataset
  - Example:



- Other application scenarios : wide application scenarios for data sharing
  - joint financial risk assessment [2], cross-platform recommendation system [3]

[2] 2024. Applying Vector Databases in Finance for Risk and Fraud Analy sis. https://zilliz.com/learn/applying-vector-databases-in-finance-for-risk-and fraud-analysis
[3] Zehua Sun, Yonghui Xu, Yong Liu, Wei He, Lanju Kong, Fangzhao Wu, Yali Jiang, and Lizhen Cui. 2025. A Survey on Federated Recommendation Systems. IEEE Trans. Neural Networks Learn. Syst. 36, 1 (2025), 6–20.

Background: motivation

- Motivation and core challenge
  - Existing methods for federated kNN search [4, 5, 6] can potentially be extended to solve this problem.
  - Limitations of efficient methods:
    - Rely on computationally expensive methods like encryption or secure multi-party computation [4, 5]
    - Only effective to low-dimensional (2D) locations or sequence data and hard to support attribute filters [6]

Core challenge: strike a balance between effectiveness and efficiency while ensuring privacy preservation

[4] Yongxin To 172.
[5] Kaining Zh
[6] Xinyi Zhan 24), V2mod011:1–V2mod011:26.

Outline

- Background
- Problem Statement
- Methodology
- Experiment
- Conclusion

Problem Statement: basic concepts

- **Vector Data**: two main components:
  - **Embedding** is denoted by a point $v.e = (e_1, e_2, ..., e_d) \in \mathrm{R}^d$
  - **Attributes** are represented by a set of � structured attributes $v.a = (a_1, a_2, ..., a_d)$
- **Attribute Filter**: P
  - represented by a conjunctive boolean predicate $P = p_1 \wedge p_2 ... \wedge p_h$
  - p_i is a binary comparison in the form $v.a_i \odot const_i, \odot \in \{ \leq, \geq, >, <, = \}$
  - $P(v) = true \leftrightarrow \forall i \in [1, h], p_i(v.a_i) = true$

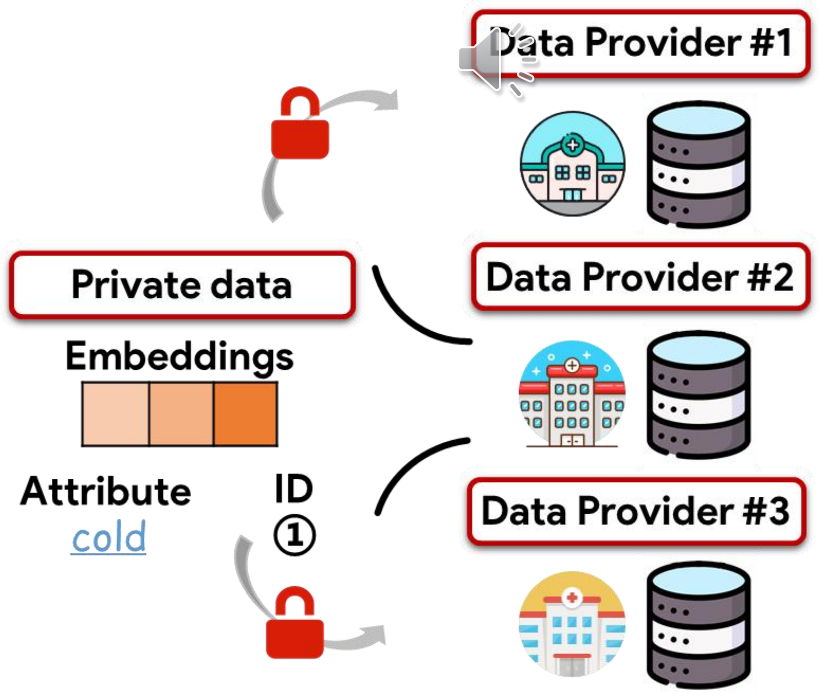
A toy example:

ID	Embedding	Drug
6	[0.2, 0, 0.1, …]	ALTO-100

Query vector : [0, 0, 0, ...], ***k*** **:** 3
Attribute filter : "Drug == ALTO-100"

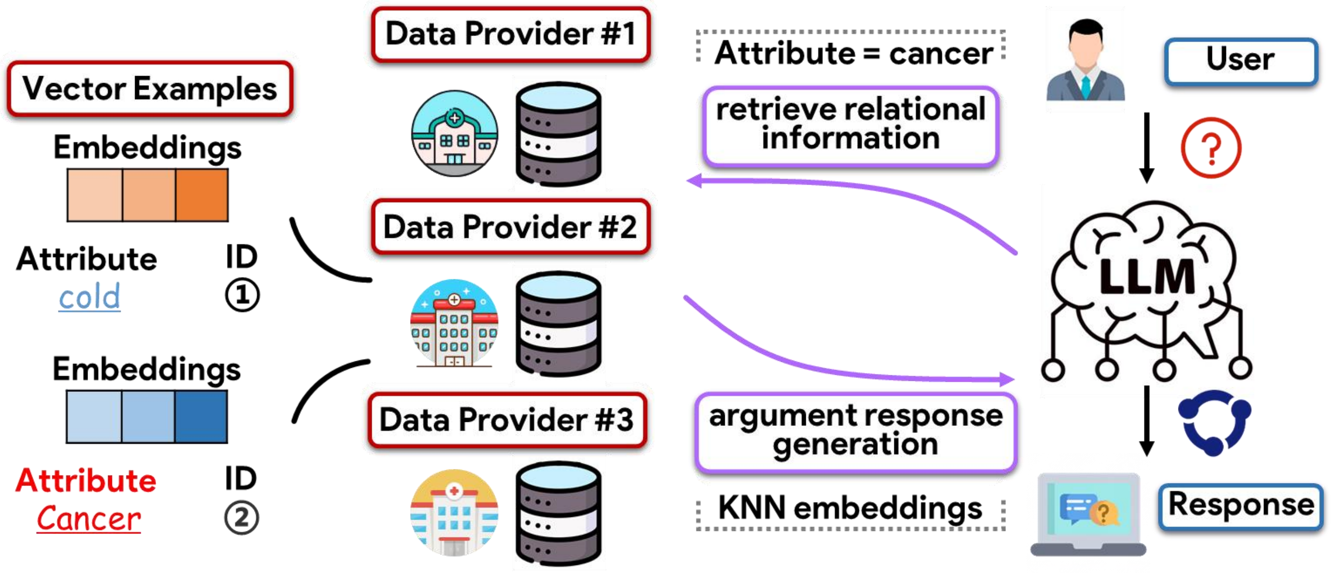
Problem Statement

- **Federated Dataset**
  - consists of � data providers, each holding a vector dataset D_i with the same data schema. All the data providers collaboratively provide vector query over $D = D_1 \cup D_2 \ldots \cup D_m$ while ensuring security



Problem Statement

- **Federated Vector Similarity Search with Filters**
  - given Federated dataset F, query (q, k, P), Res meets the following two constraints:
    - **Filter constraint**: $\forall v \in Res,\ P(v) = true$
    - **kNN constraint**: D^{-} denotes the set of vectors satisfying the filter constraint.
  - **Security constraints**: query user only learns results, data providers cannot access or defer others' private data.



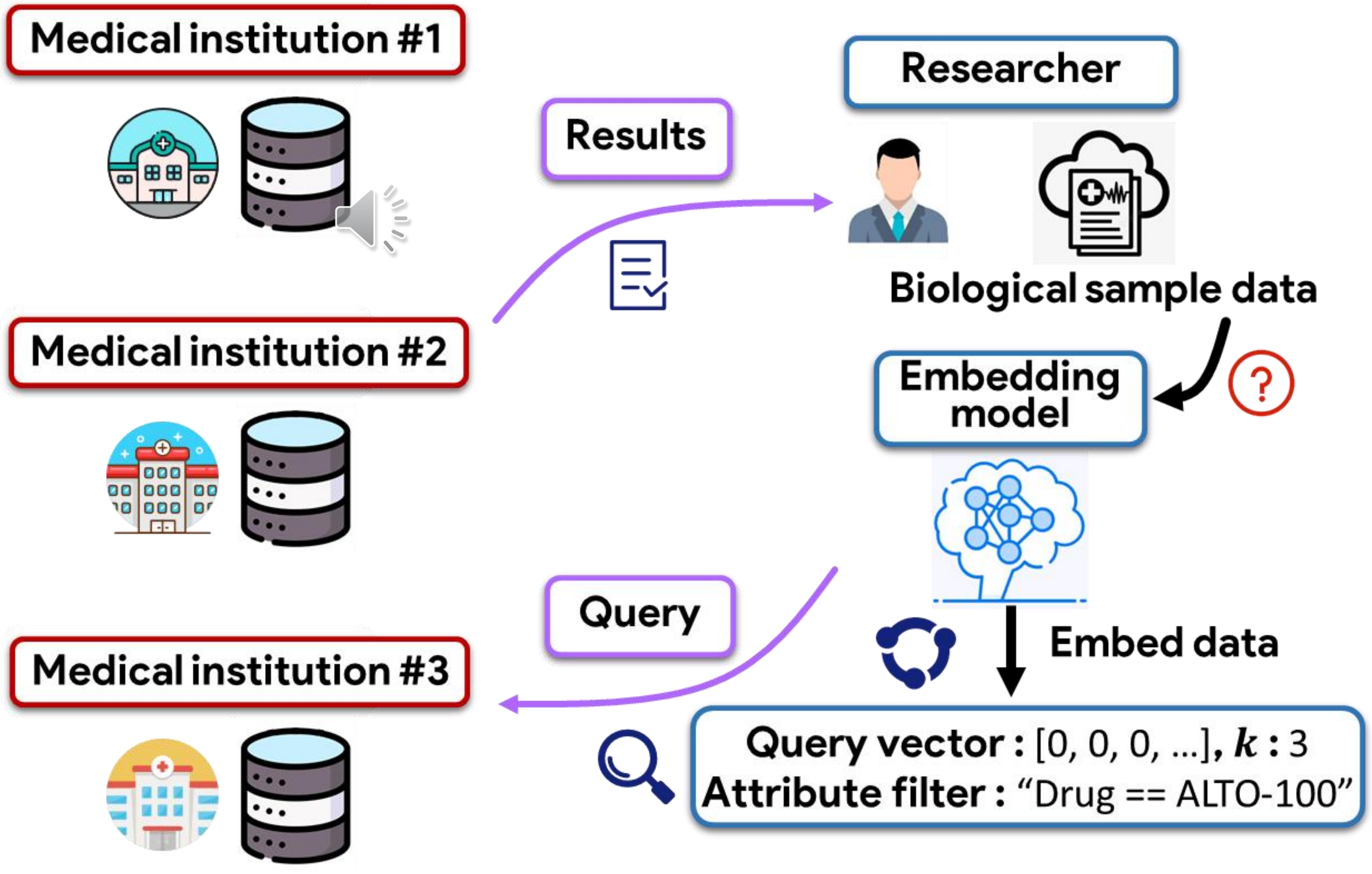
Problem Statement: a toy example

- Application scenario for **collaborative pharmaceutical development**
  - Query vector: [0, 0, 0, 0] k: 3 Attribute filter: Drug == ALTO-100

ID	Embedding	Drug
6	[0.2, 0, 0.1, ...]	ALTO-100
7	[0.1, 0, 0.1, ...]	5-MAPB
8	[0.2, 0.3, 0.3, ...]	ALTO-100

ID	Embedding	Drug
3	[0.1, 0.1, 0.1, …]	ALTO-100
4	[0, 0.1, 0.1, …]	ALTO-100
5	[0.2, 0.1, 0.2, …]	ALTO-100

ID	Embedding	Drug
1	[0.1, 0.2, 0, ...]	5-MAPB
2	[0, 0, 0.1, ...]	ALTO-100

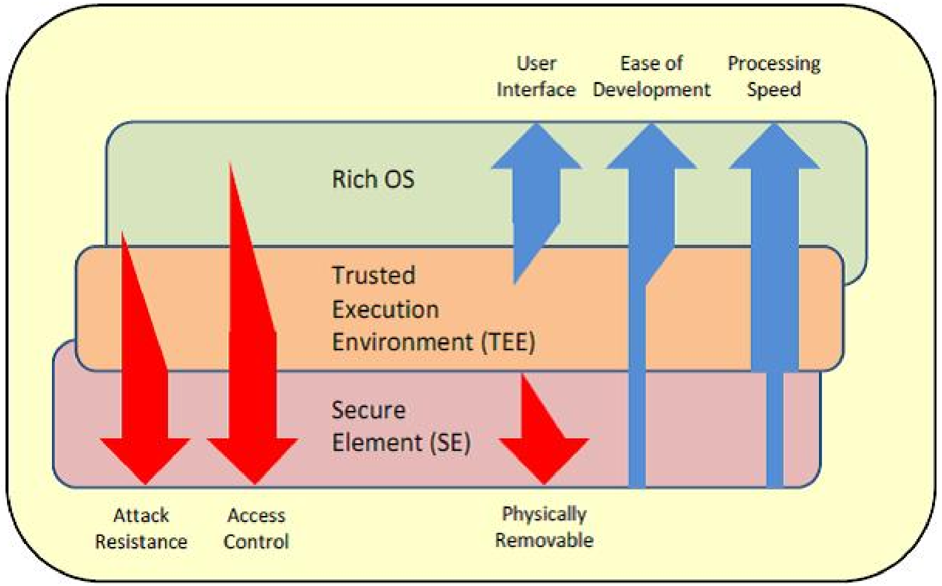


Outline

- Background
- Problem Statement
- Methodology
- Experiment
- Conclusion

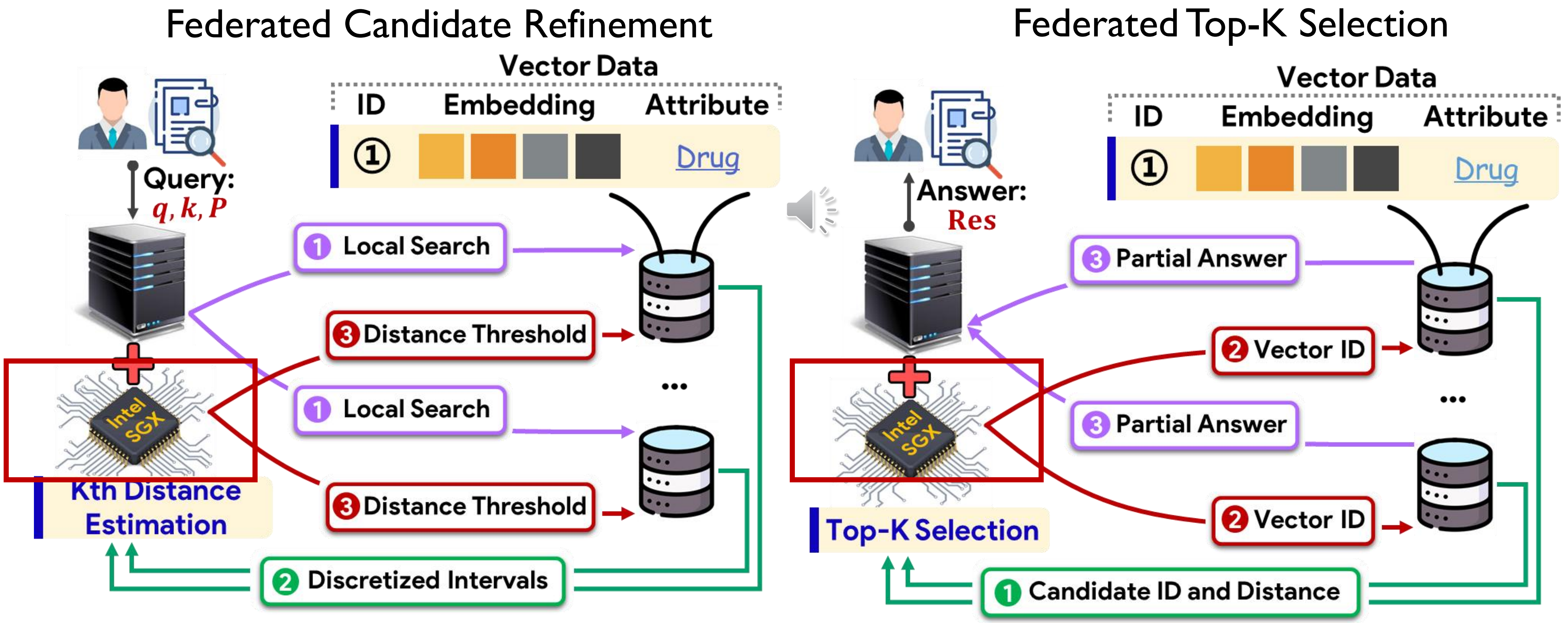
Methodology: a brief introduction to TEE

- Trusted Execution Environment (TEE)
  - A hardware-assisted technique for privacy preserving
  - TEE offers a secure and isolated area within the CPU and memory, where private data can be processed with strong confidentiality guarantees
  - TEE encrypts and authenticates private data, ensuring only the same enclave can decrypt it.



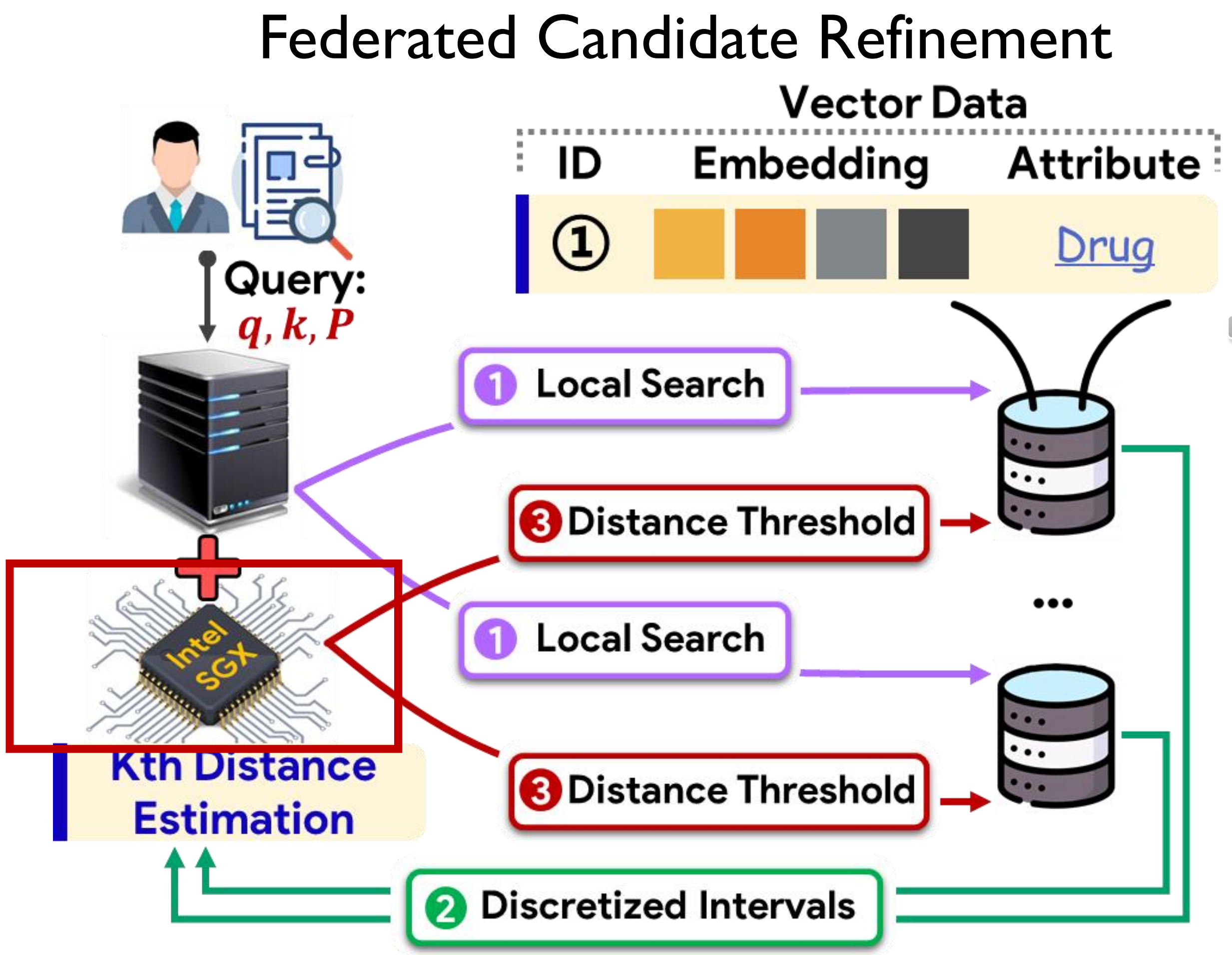
Methodology: framework

- FedVS: two-phase framework based on TEE



Methodology: framework

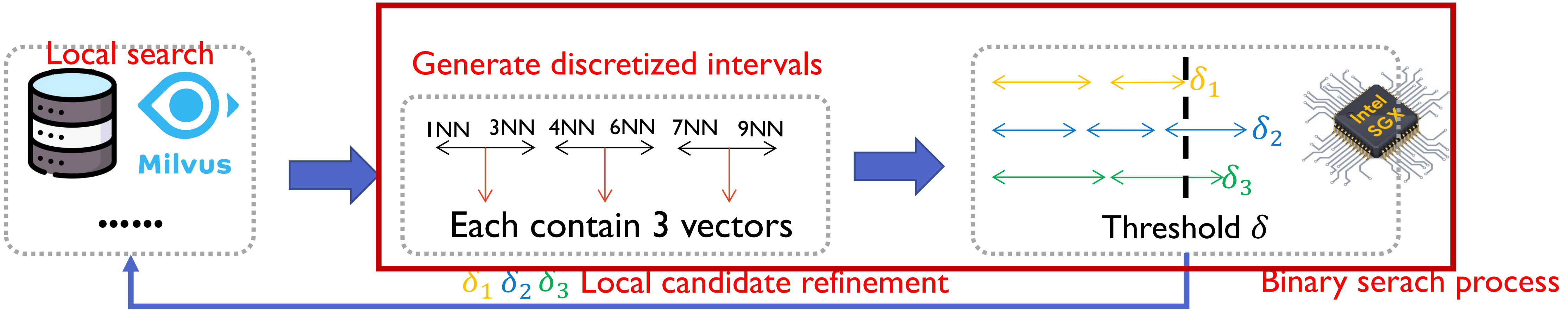
- FedVS: two-phase framework based on TEE



- Phase I: derive a threshold for upper bound of the Kth nearest distance
  - Data Provider: Local search and discretize candidates into $\sqrt{k}$ intervals
  - Central Server (TEE): calculate threshold with binary search based on intervals

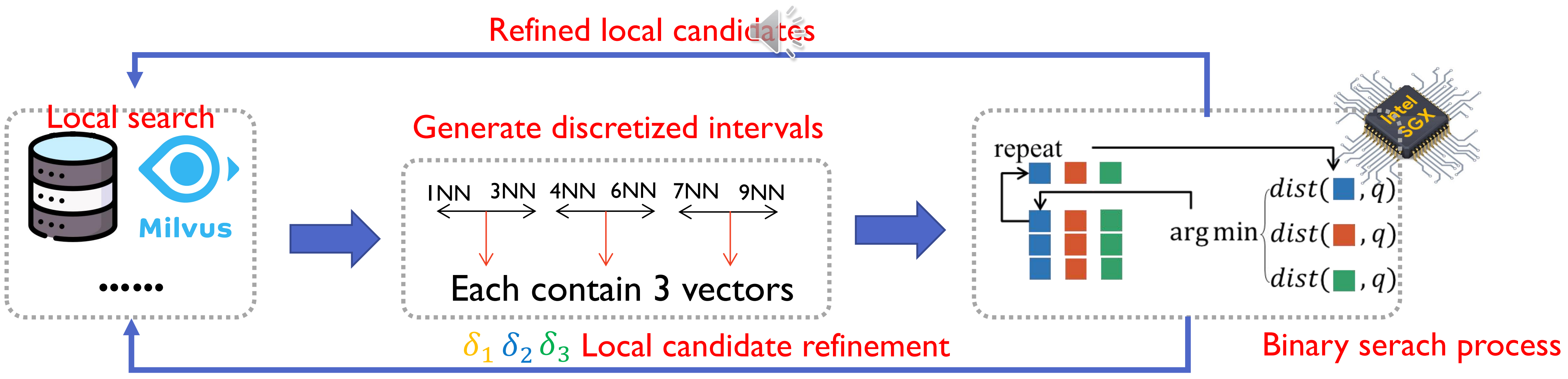
Methodology: framework

- FedVS: two-phase framework based on TEE
  - Phase I: derive a threshold for upper bound of the Kth nearest distance
    - Data Provider: Local search and discretize candidates into $\sqrt{k}$ intervals
    - Central Server (TEE): calculate threshold with binary search based on intervals



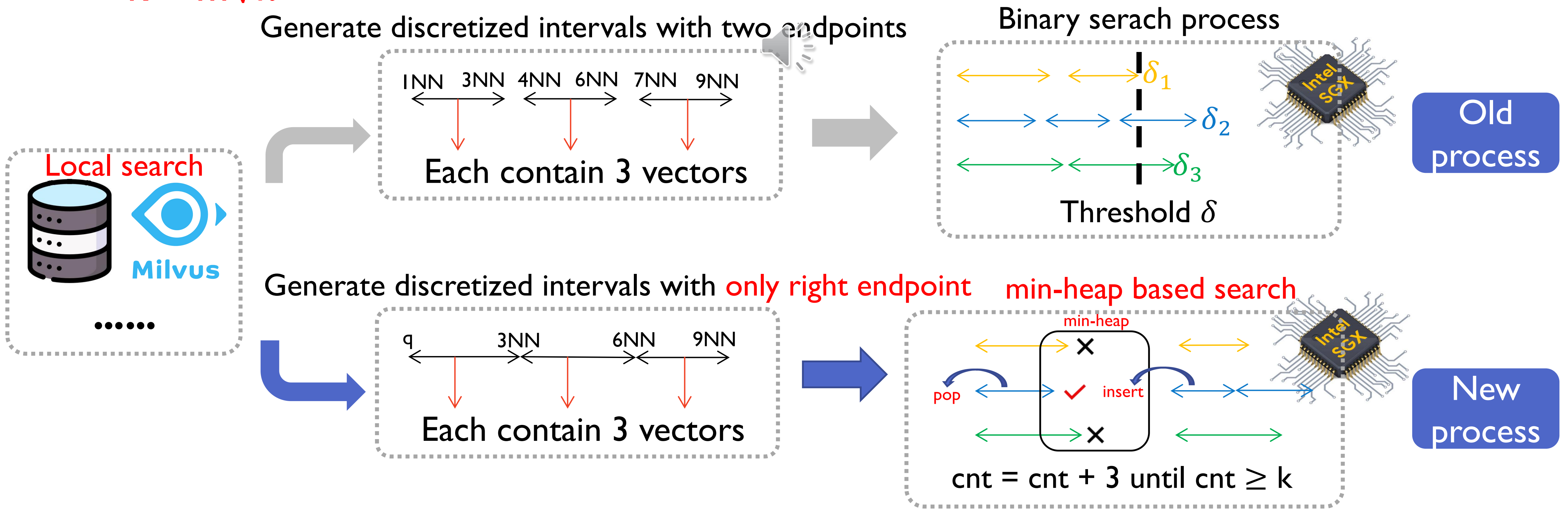
Methodology: framework

- Phase II: select top-k from refined candidates
  - Data Provider: eliminate candidates with distance larger than threshold
  - Central Server (TEE): top-k selection with an (oblivious) priority queue



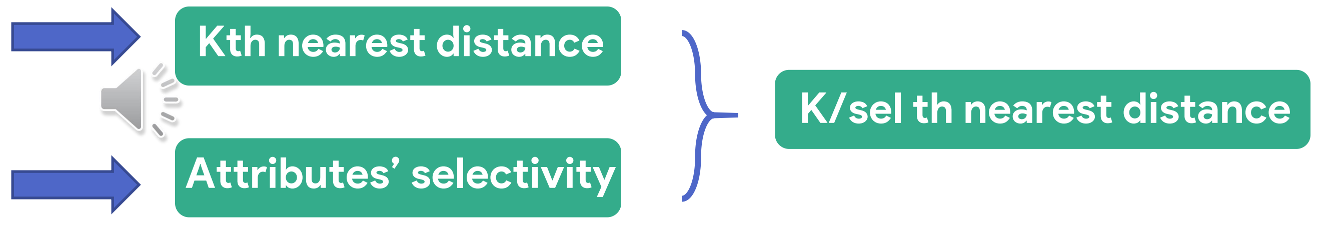
Methodology: optimization #1

- Optimization I: reducing communication cost
  - Simplified representation for intervals: We only care the right endpoint !
  - Replacing binary search with min-heap based search: Bound number of candidates to $k + m\sqrt{k}$



Methodology: optimization #2

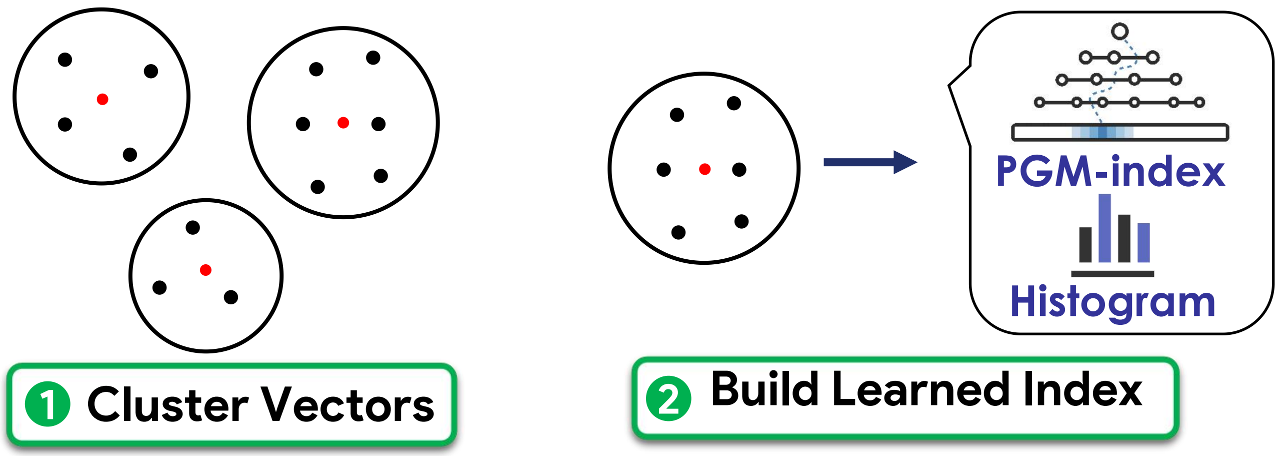
- Optimization II: pruning via contribution estimation
  - Motivation: We don’t need to select k candidates from each provider
  - Eliminate local redundant candidates with two primary factors:
    - Distance to query vector
    - Selectivity of attribute filter



Methodology: optimization #2

- Optimization II: pruning via contribution estimation
  - We estimate contribution through a proposed Cluster-based Learned Index (CLI)
  - It takes two steps to build our CLI

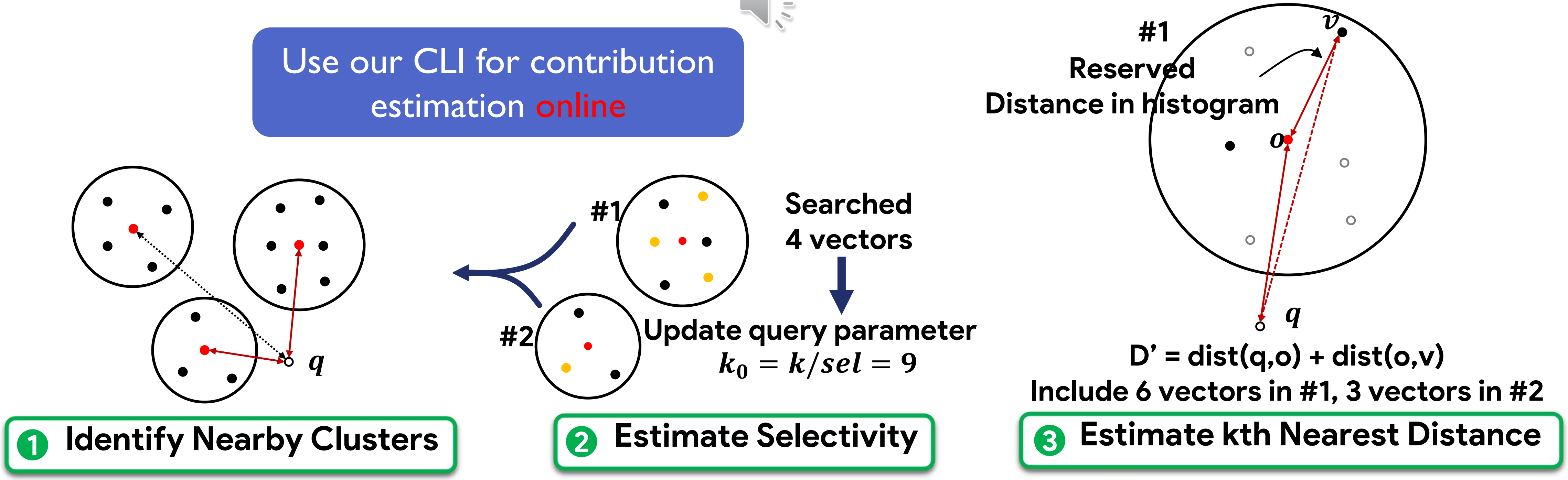
Build our CLI offline



Methodology: optimization #2

- Optimization II: pruning via contribution estimation
  - Each provider builds CLI offline
  - Each provider estimates distance of the K/sel th nearest neighbor to q among nearby clusters as own contribution online

Use our CLI for contribution estimation online


#1
Reserved
Distance in histogram
v
o
q
#1
#2
Searched
4 vectors
Update query parameter
$k_0 = k/sel = 9$
D' = dist(q,o) + dist(o,v)
Include 6 vectors in #1, 3 vectors in #2
1 Identify Nearby Clusters
2 Estimate Selectivity
3 Estimate kth Nearest Distance

Methodology: optimization #2

- Optimization II: pruning via contribution estimation
  - Each provider calculates distance γ_i^*, then submits it to TEE
  - TEE calculates a pruned k for each data provider through:

$$k_i = \frac{k * \min_i \gamma_i^*}{\gamma_i^*}$$

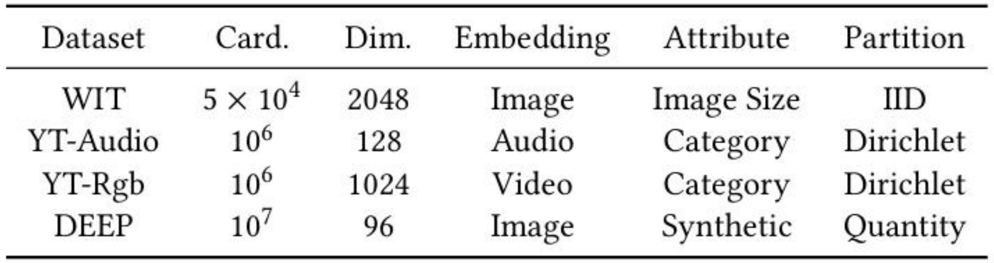
  - The above formula indicates that providers with smaller γ_i^* are likely to contribute more significantly in the final answer
  - Moreover, the provider with the minimum γ_i^* remains with � initial candidates to retain high recall

Outline

- Background
- Problem Statement
- Methodology
- Experiment
- Conclusion

Experiment: setup

- Dataset
  - WIT, YT-Audio, TY-Rgb, Deep
  - with at most 2048 dimensionality, 10e7 cardinary and two structured attributes over both NON-IID and IID vector data



Dataset	Card.	Dim.	Embedding	Attribute	Partition
WIT	5×10^4	2048	Image	Image Size	IID
YT-Audio	10^6	128	Audio	Category	Dirichlet
YT-Rgb	10^6	1024	Video	Category	Dirichlet
DEEP	10^7	96	Image	Synthetic	Quantity

- Deployment
  - We deploy our experimental study on six cloud servers over industrial vector databases Milvus [7]
  - The main hardware includes Intel Xeon(R) Platinum 8361HC CPUs and 32GB of RAM, with one server equipped with Intel's SGX SDK

[7] 2024. Milvus. https://milvus.io

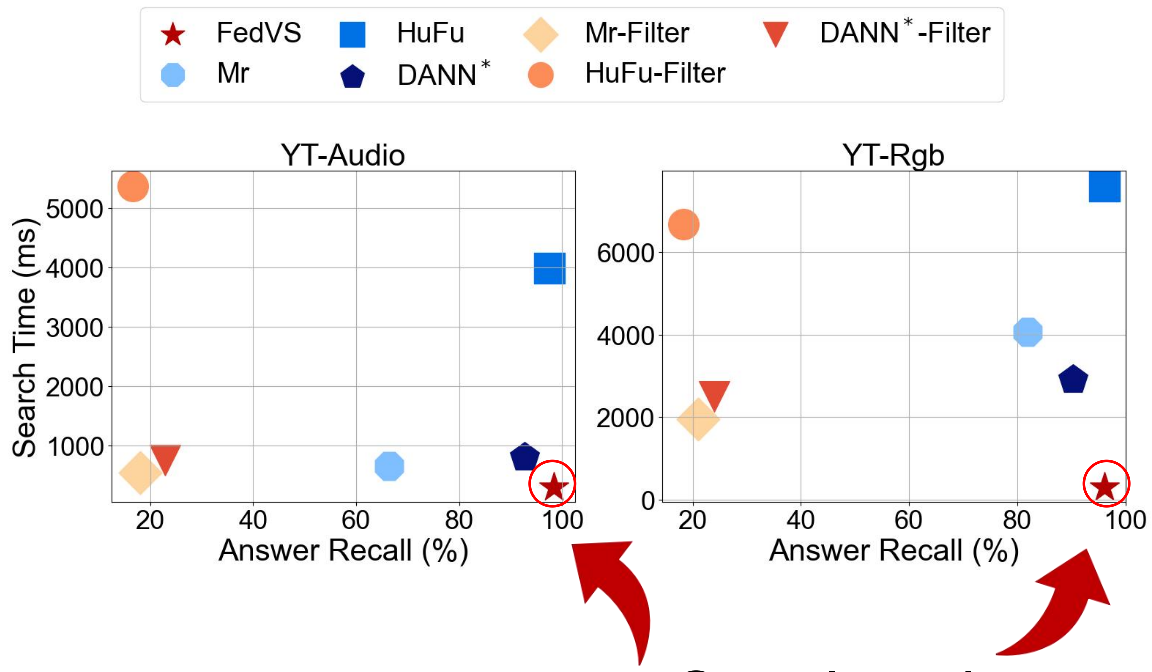
Experiment: setup

- Compared Algorithms (with extensions for attribute filter and high-dimensional vectors)
  - **HuFu [4]**: federated search over 2D vectors with Secure summation and secure comparison
  - **Mr [5]**: federated search over 2D vectors with multiple rounds of contribution estimation and secure summation.
  - **DANN* [6]**: federated vector search with TEE and Differential Privacy
  - **(HuFu, Mr, DANN*)-Post**: conduct local search with a "post-filter" strategy

[4] Yongxin Tong, Xuchen Pan, Yuxiang Zeng, et al. 2022. Hu-Fu: Efficient and Secure Spatial Queries over Data Federation. PVLDB 15, 6 (2022), 1159–1172.
[5] Kaining Zhang, Yongxin Tong, Yexuan Shi, et al. 2023. Approximate k-Nearest Neighbor Query over Spatial Data Federation. In DASFAA. 351–368. **best paper**
[6] Xinyi Zhang, Qichen Wang, Cheng Xu, Yun Peng, and Jianliang Xu. 2024. FedKNN: Secure Federated k-Nearest Neighbor Search. SIGMOD 2, 1 (2024), V2mod011:1–V2mod011:26.

Experiment: overall query performance

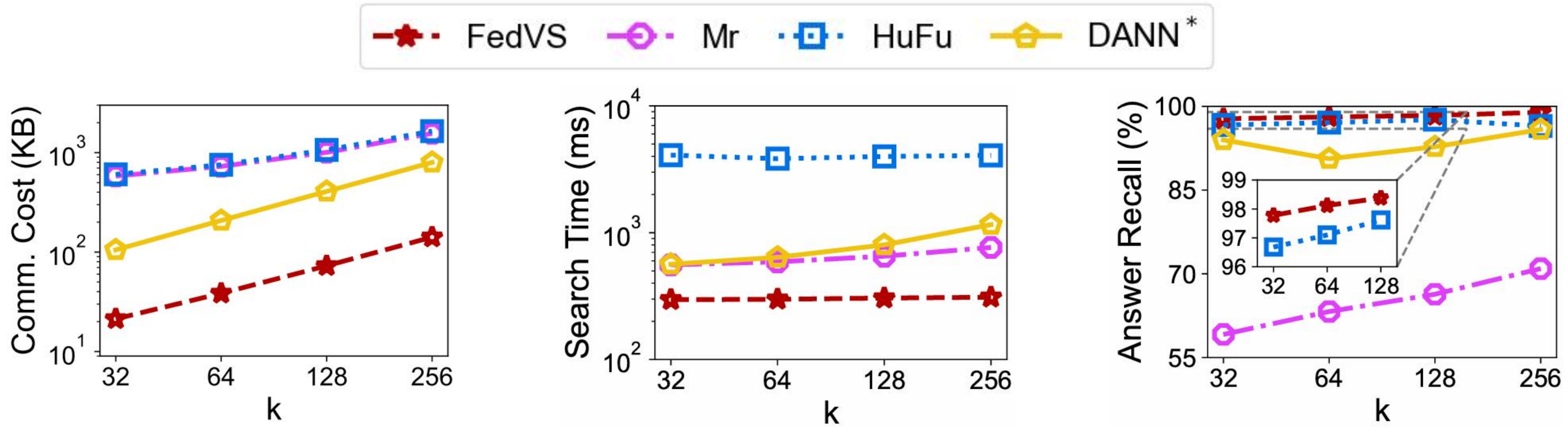
- Default exp parameters:
  - Query parameter k is 128, #(data providers) � is 5.
- Overall performance:
  - Our solution(FedVS) achieves best query Efficiency (at most 27.25× lower search time and 15.32× lower communication cost)
  - Our solution(FedVS) consistently achieves the highest recall (up to 32.03% higher than HuFu, Mr and DANN*)



Our algorithm

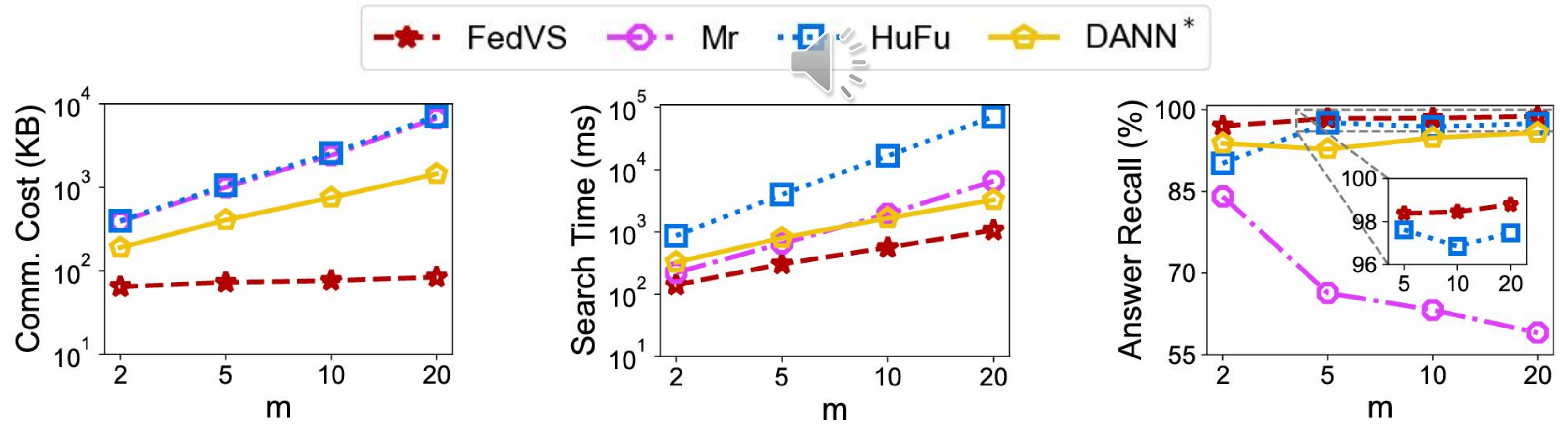
Experiment: impact of query parameters

- Vary query parameter k from 32~256:
  - FedVS still achieves best performance among all values of k, with up to 25.18× faster



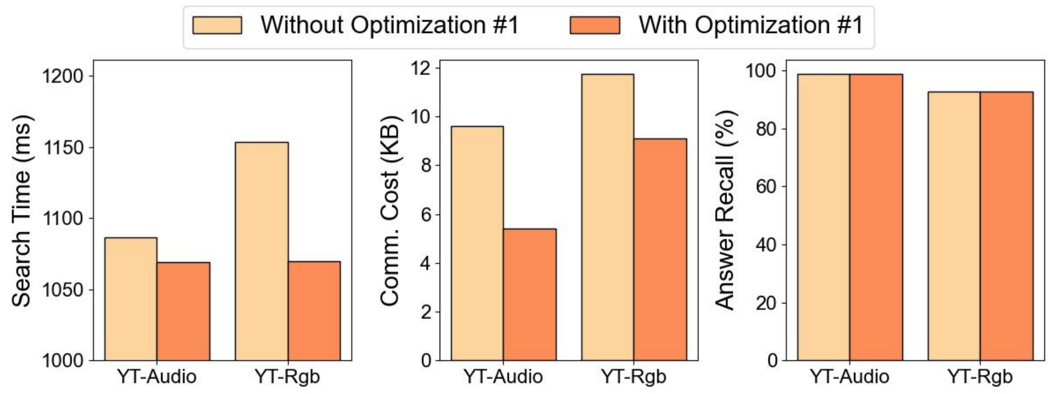
Experiment: impact of query parameters

- Vary #(data providers) m from 2~20:
  - FedVS always maintains more stable communication / time overhead and answer recall



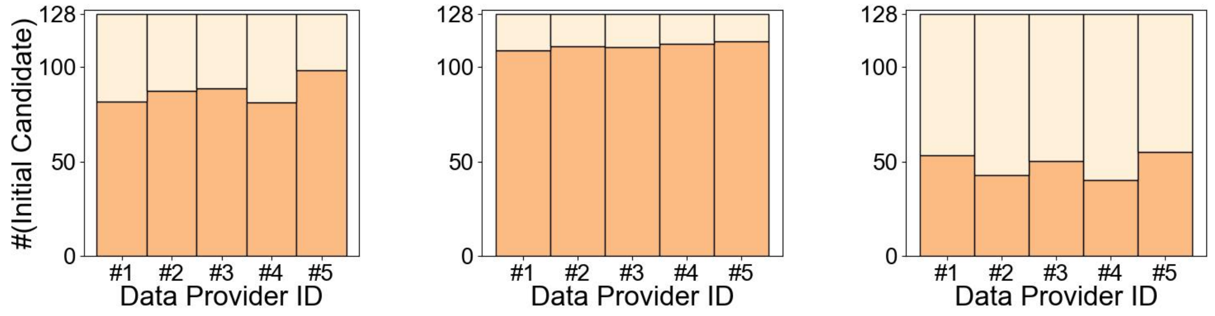
Experiment: ablation study

- Ablation study on optimization #1:
  - FedVS effectively reduces both the search time and communication overhead by 7.32% and 22.33% while maintaining accuracy.



- Ablation study on optimization #2:
  - Optimization #2 can reduce the initial candidate size at each provider by up to 15.19%–68.56% with less than 2 seconds and 1MB space to build our CLI

Dataset	YT-Audio	YT-Rgb	DEEP
Clustering Time	28s	150s	6778s
Index Build Time	31ms	18ms	1789ms
Index Space Cost	17KB	51KB	346KB



YT-Audio YT-Rgb DEEP

Outline

Conclusion

- Our work introduces a new problem called federated vector similarity search with filters
- We propose a two-phase framework FedVS based on TEE and devise two optimizations via indexing and pruning.
- Overall, FedVS accelerates search time by up to 27.25× and reduces communication overhead by up to 15.32× while keeping high recall
- Our source code and real data are now available at https://doi.org/10.5281/zenodo.15504203 and welcome to

contact me by email: fanzh@buaa.edu.cn

Thank you for listening