Federated Retrieval over Embedding-Heterogeneous Vector Databases

Yuxiang Wang [1], Yongxin Tong [1], Zimu Zhou [2],
Ziyuan He [1], Ruixi Hu [1], Ke Xu [1]

[1] Beihang University

[2] City University of Hong Kong

Outline

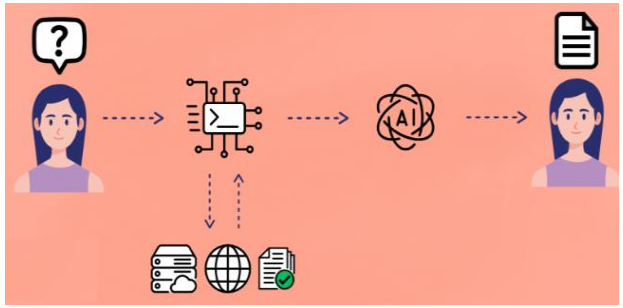
- Background
- Problem Statement
- Method
- Experiment
- Conclusion

Background

- Widespread Applications of Vector Database

Recommendation

Multimodal Search

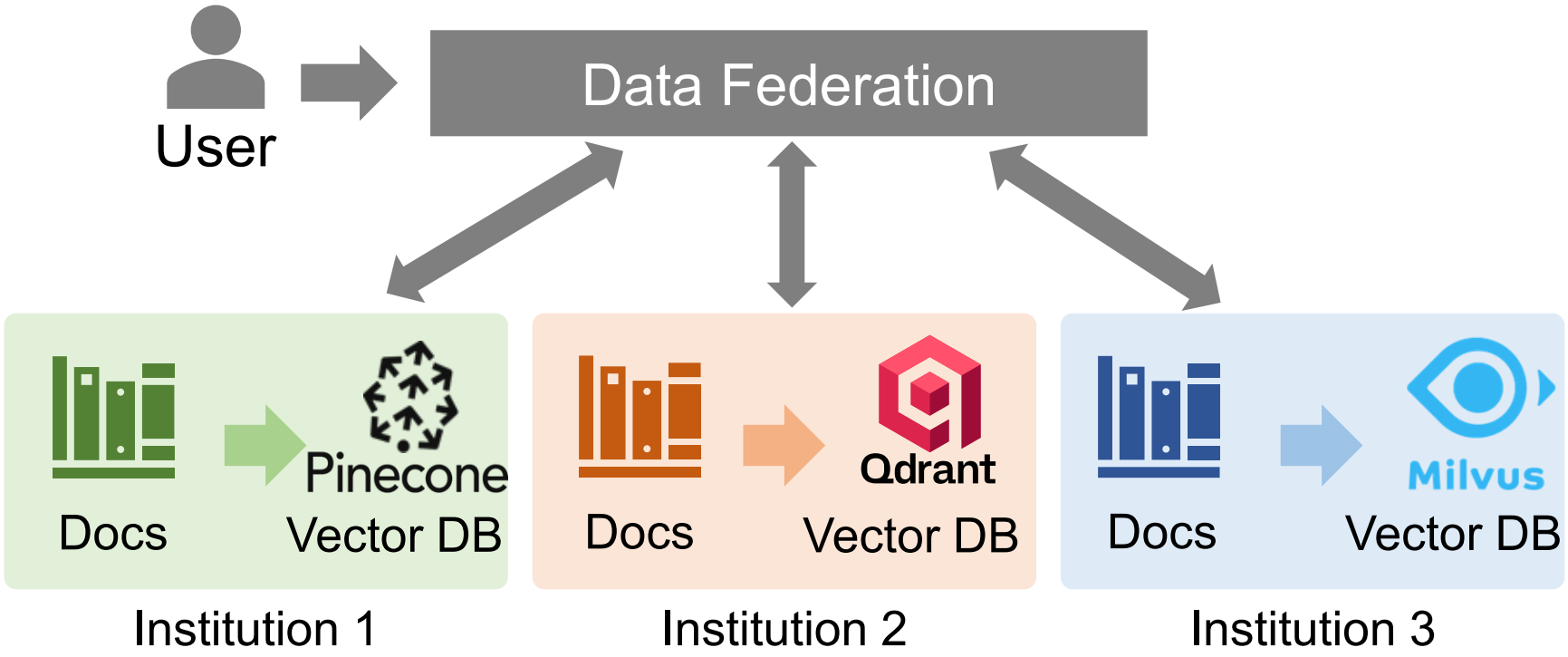


RAG

Agent Memory

Background

- Scaling Vector Search to **Data Federation**
  - Example: Question answering with **multi-source** scientific document corpus
  - Each institution **autonomously** manages a vector database for its local documents



Outline

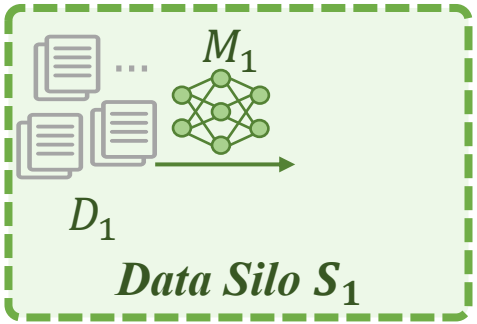
- Background
- Problem Statement
- Method
- Experiment
- Conclusion

Problem Definition

- Data silo S_i
  - Hold a dataset of unstructured data object $D_i = \{o_i^1, o_i^2, \dots o_i^{|D_i|}\}$

Problem Definition

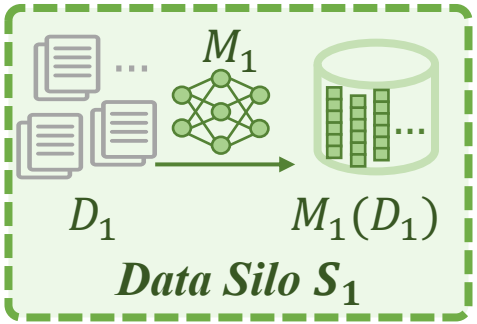
- Data silo S_i
  - Hold a dataset of unstructured data object $D_i = \{o_i^1, o_i^2, \ldots o_i^{|D_i|}\}$
  - Use silo-specific embedding model M_i



Problem Definition

- Data silo S_i
  - Hold a dataset of unstructured data object $D_i = \{o_i^1, o_i^2, \dots o_i^{|D_i|}\}$
  - Use silo-specific embedding model M_i
  - Operates a vector database over $M_i(D_i)$ to achieve efficient search

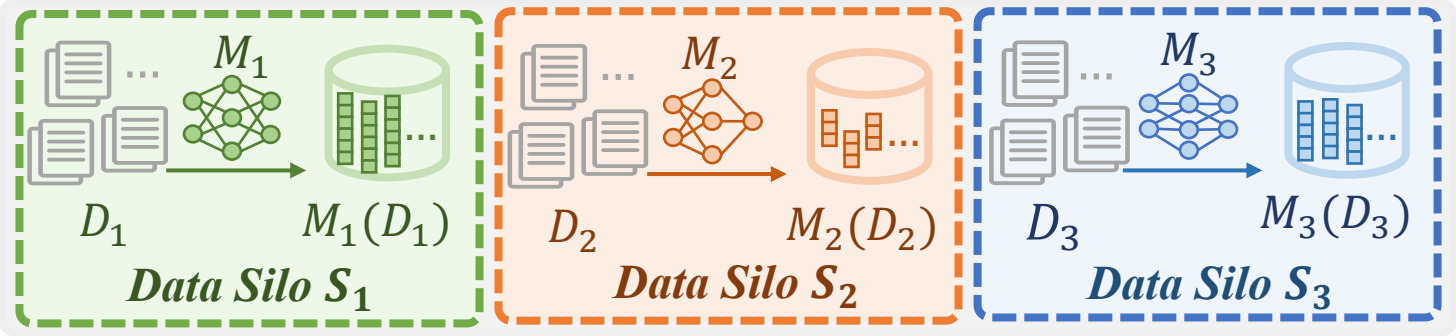
We denote by $M_i(\cdot)$ the high-dimensional vector produced by model M_i



Problem Definition

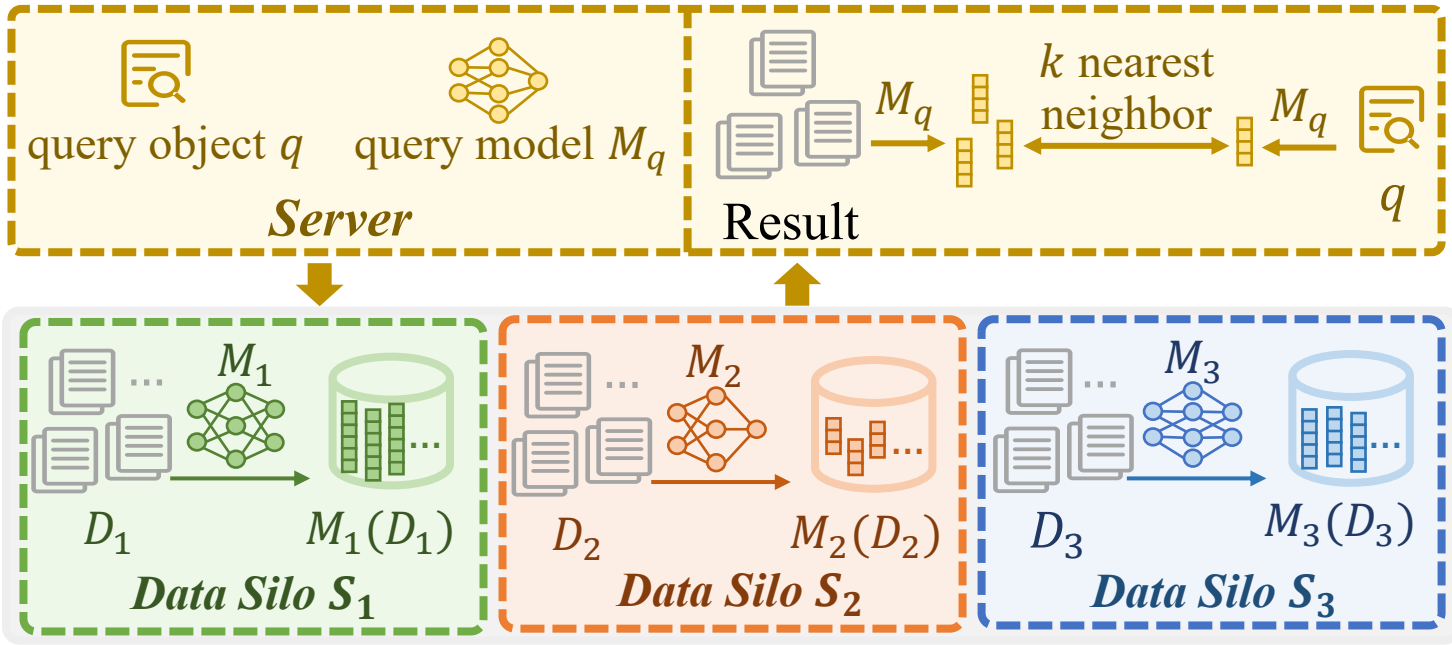
- Federated Approximate Nearest Neighbor Search (FANNS)
  - Operates on data federation consisting of n data silos

Each silo may adopt different embedding model (**Embedding Heterogeneity**)



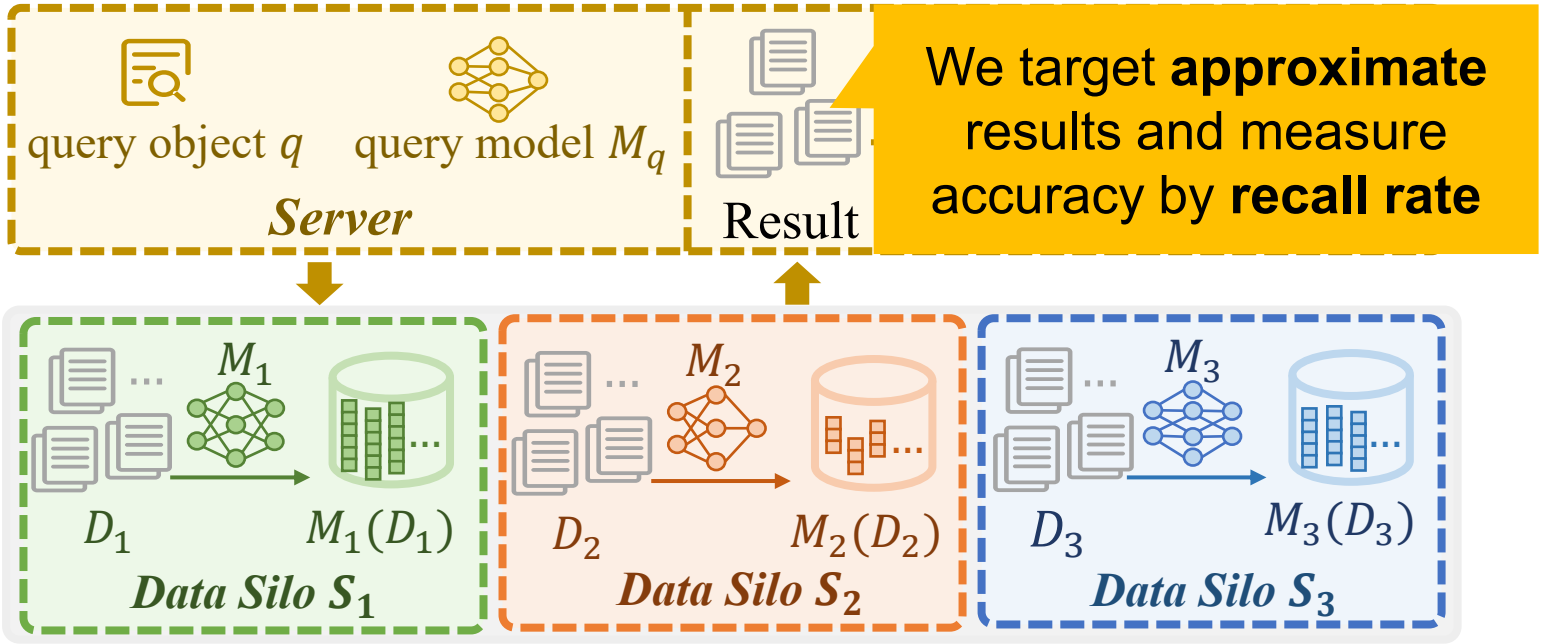
Problem Definition

- Federated Approximate Nearest Neighbor Search (FANNS)
  - Operates on data federation consisting of n data silos
  - Given a query object q and a query model M_q
  - Retrieves k objects from $D = D_1 \cup D_2 \cup \cdots \cup D_n$ that are nearest to q in the embedding space of M_q



Problem Definition

- Federated Approximate Nearest Neighbor Search (FANNS)
  - Operates on data federation consisting of n data silos
  - Given a query object q and a query model M_q
  - Retrieves k objects from $D = D_1 \cup D_2 \cup \cdots \cup D_n$ that are nearest to q in the embedding space of M_q



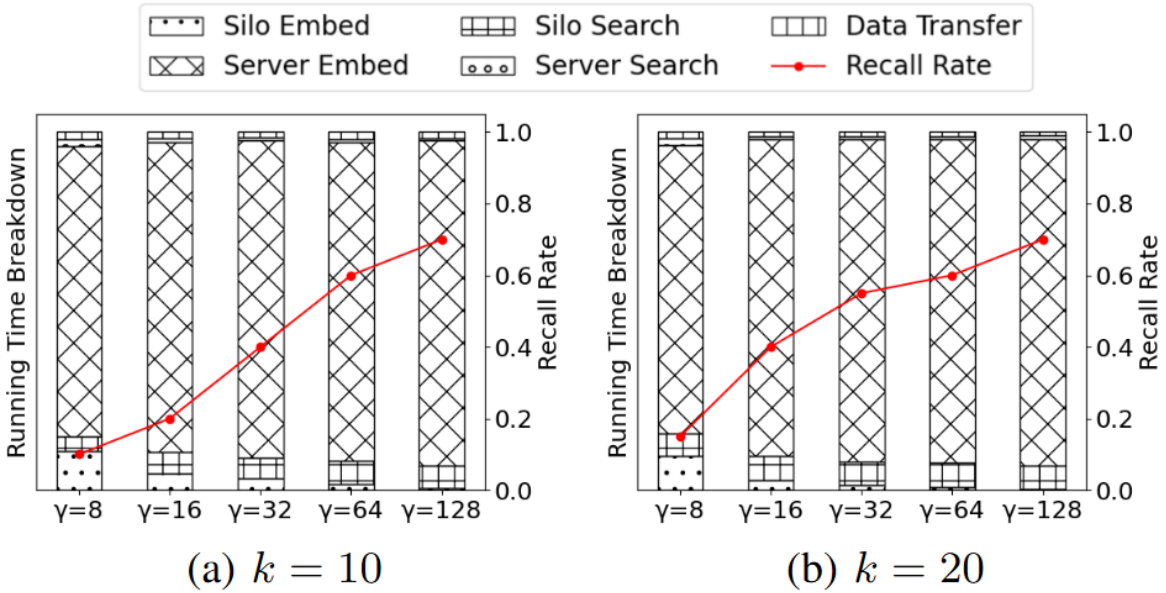
Challenges

- Naïve Solution: Uniform Selection Algorithm
  - Each data silo returns the same number of objects, which are reranked by server using the query model

Challenges

- Naïve Solution: Uniform Selection Algorithm
  - Each data silo returns the same number of objects, which are reranked by server using the query model
  - Observation #1: Server re-embedding dominates latency (≥90%)
  - Observation #2: High recall rate demands re-embedding a large amount of objects (k objects per silo insufficient)

Runtime Breakdown

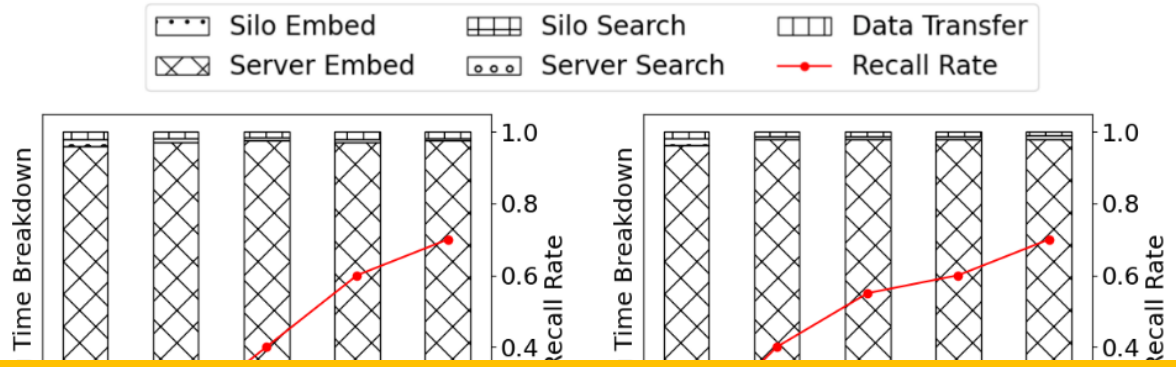


(a) $k = 10$ (b) $k = 20$

Challenges

- Naïve Solution: Uniform Selection Algorithm
  - Each data silo returns the same number of objects, which are reranked by server using the query model
  - Observation #1: Server re-embedding dominates latency (≥90%)
  - Observation #2: High recall rate demands re-embedding a large amount of objects (k objects per silo insufficient)

Runtime Breakdown



Opportunities: Dynamically adjusting per-silo retrieval sizes to reduce re-embedding operations

Outline

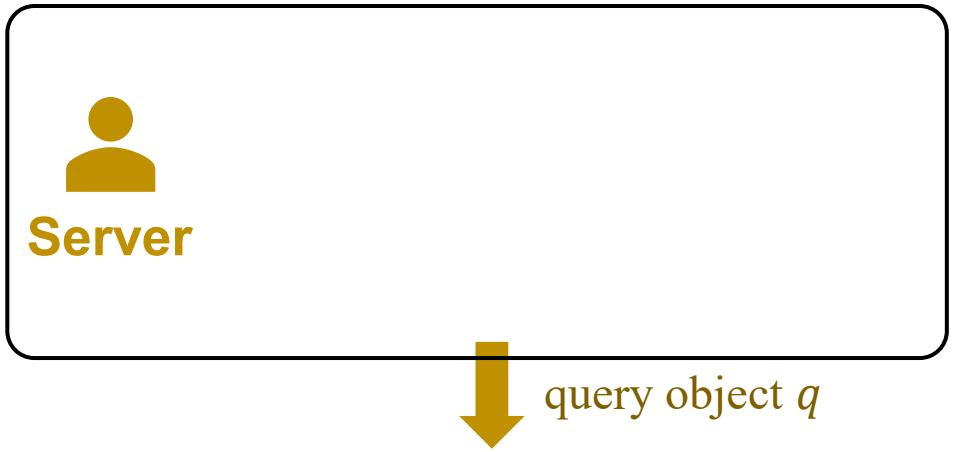
- Background
- Problem Statement
- Method
- Experiment
- Conclusion

Competition-based FANNS

- Basic Idea: multi-round competition that prioritizes silos providing the most relevant object in each round
  - Each silo proposes its top candidate based on its local embedding model M_i
  - Server re-embed candidates under embedding model M_q
  - Server request the silo whose candidate is closest (the “winner”) to provide its next-best candidate

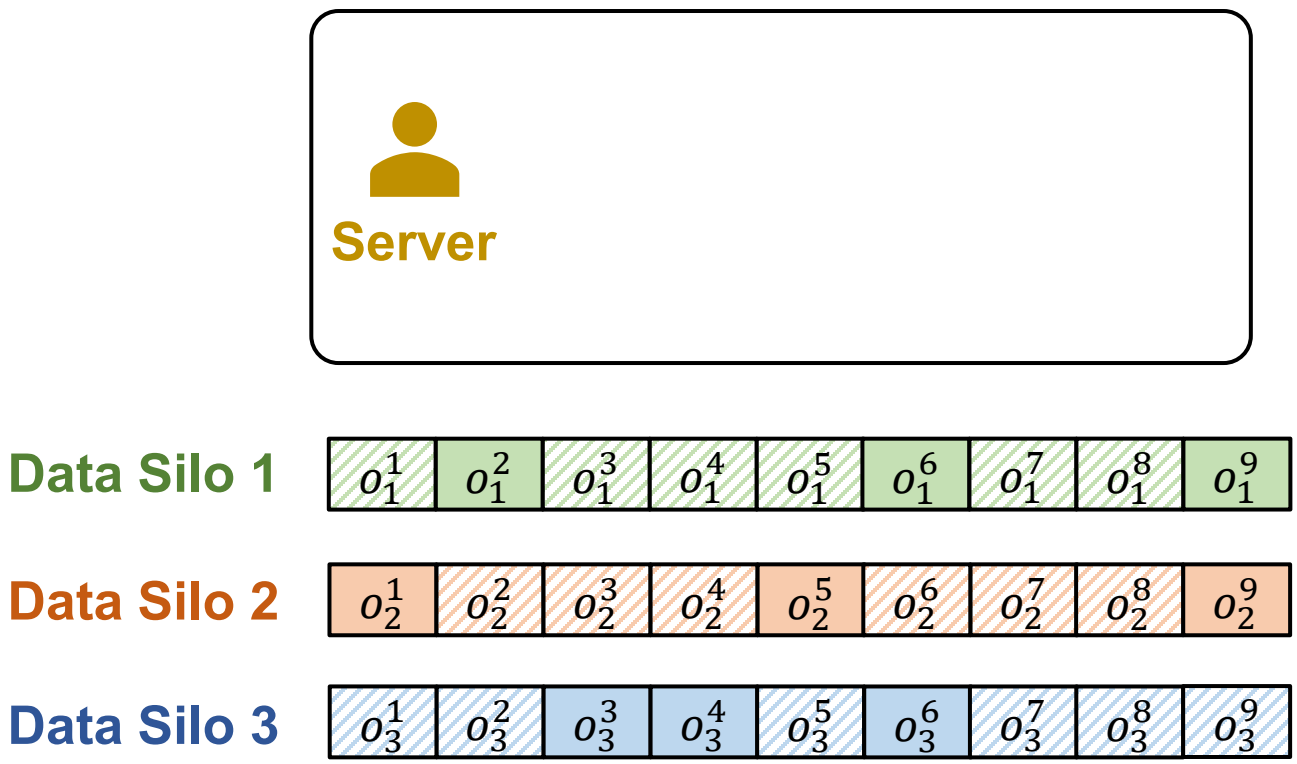
Competition-based FANNS

- Running Example ($k = 3$)
  - Server broadcast query object to data silos



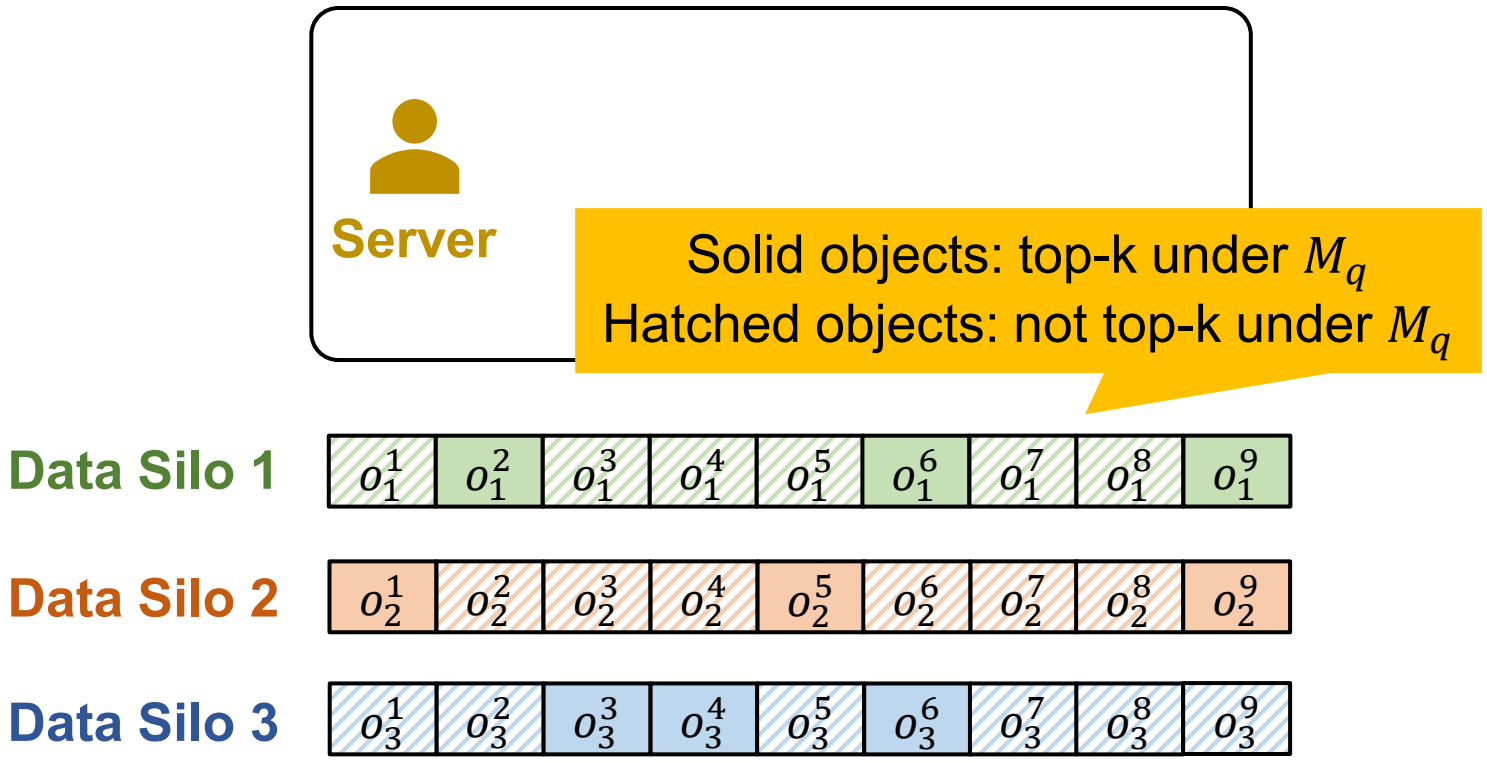
Competition-based FANNS

- Running Example ($k = 3$)
  - Each silo selects candidates based on distance in local embedding space



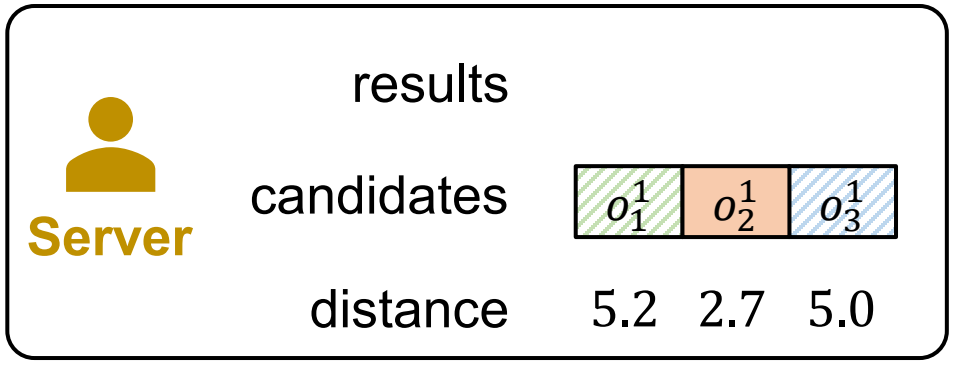
Competition-based FANNS

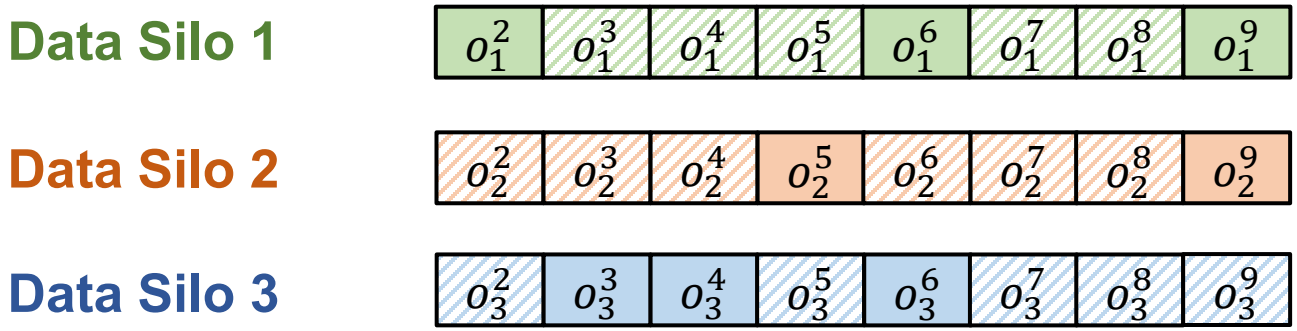
- Running Example ($k = 3$)
  - Each silo selects candidates based on distance in local embedding space



Competition-based FANNS

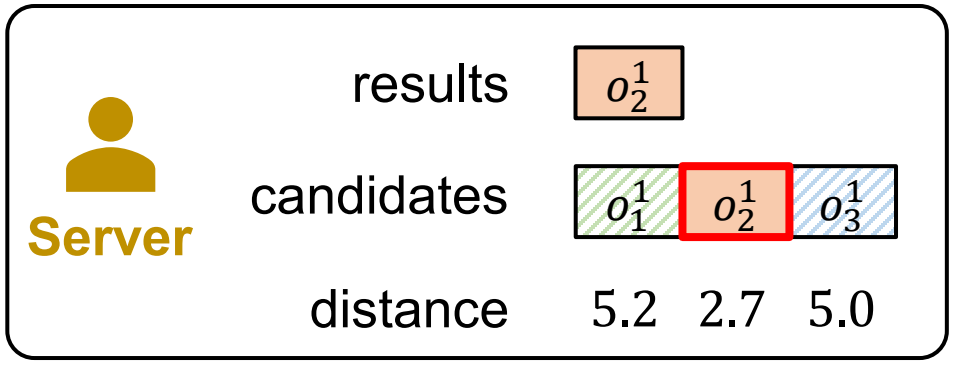
- Running Example ($k = 3$)
  - Each silo proposes one candidate

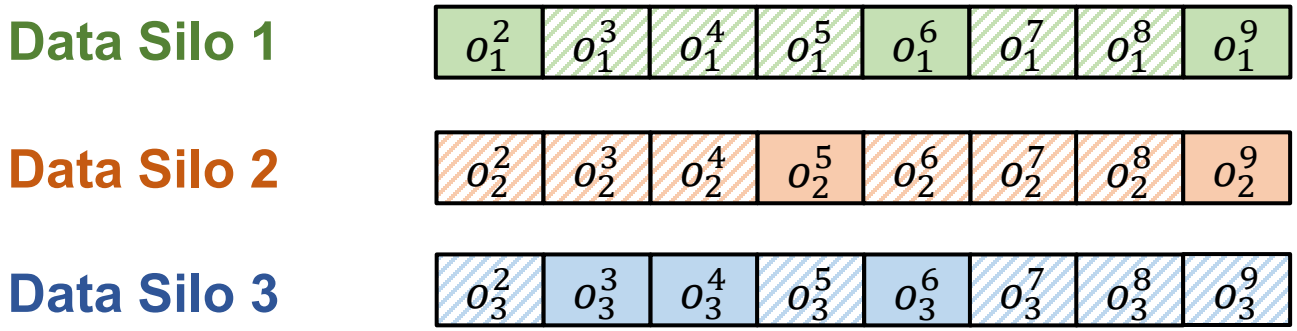




Competition-based FANNS

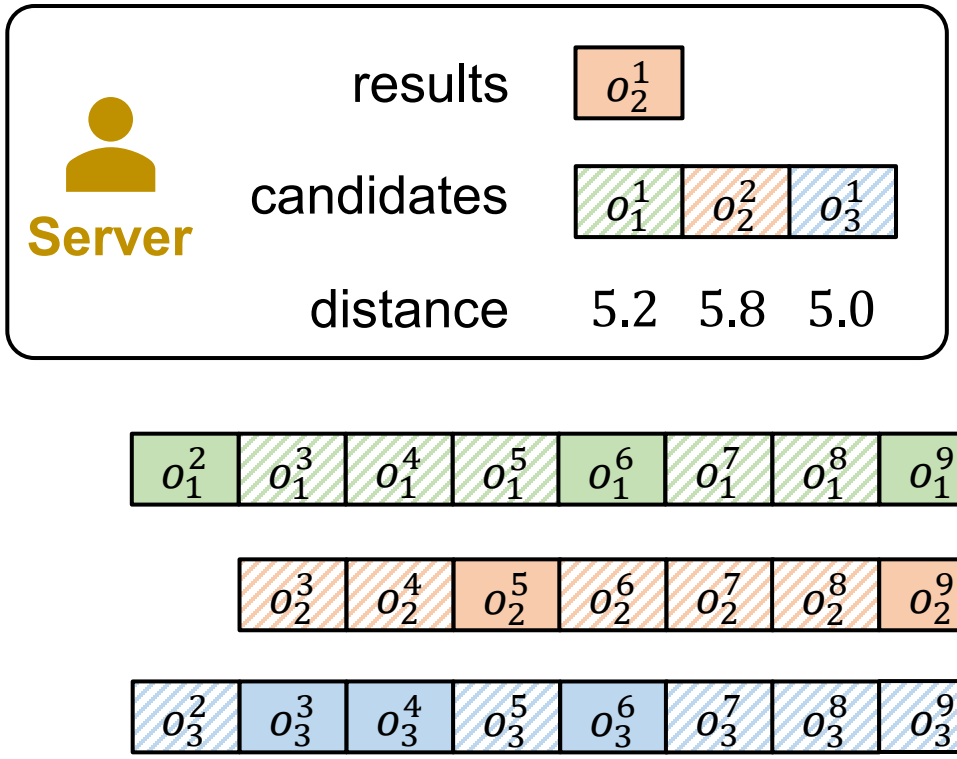
- Running Example ($k = 3$)
  - Select the best candidate o_2^1 into result





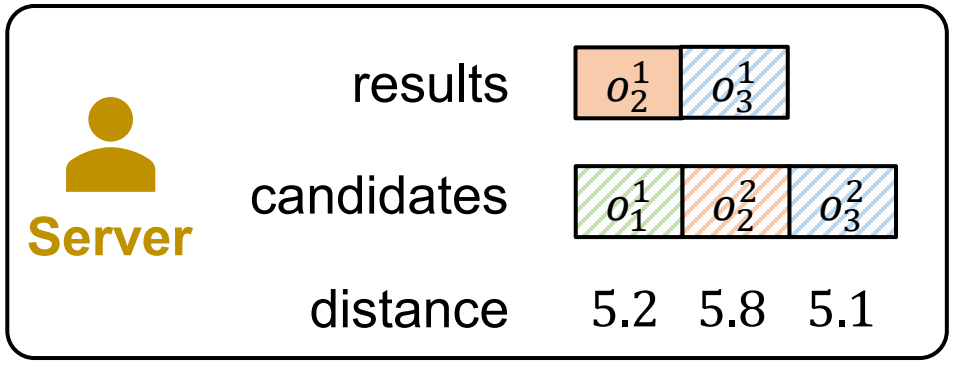
Competition-based FANNS

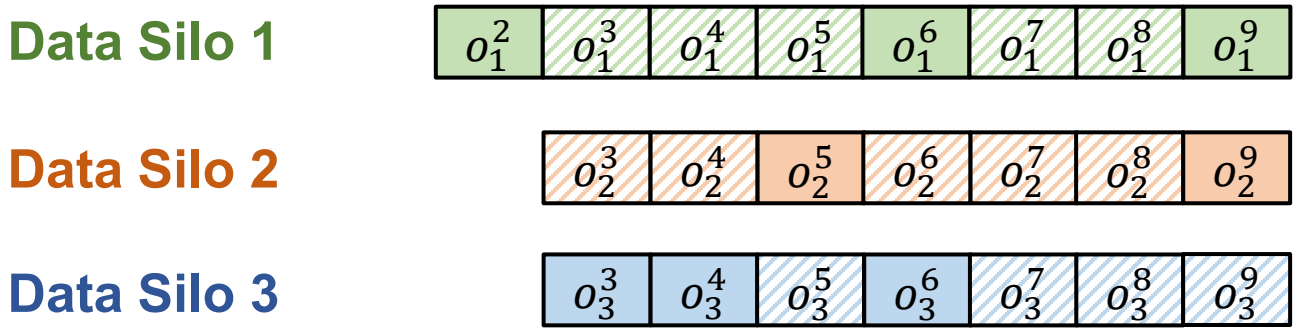
- Running Example ($k = 3$)
  - Require silo 2 to provide next candidate



Competition-based FANNS

- Running Example ($k = 3$)
  - Select o_3^1 into result and require silo 3 to provide next candidate





Contribution-based FANNS

- Basic Idea: select silos according to their cumulative contributions over previous rounds
  - Specifically, if silo S_i has contributed t_i objects to the current top-k results, its sampling probability is set proportional to $t_i + \theta$

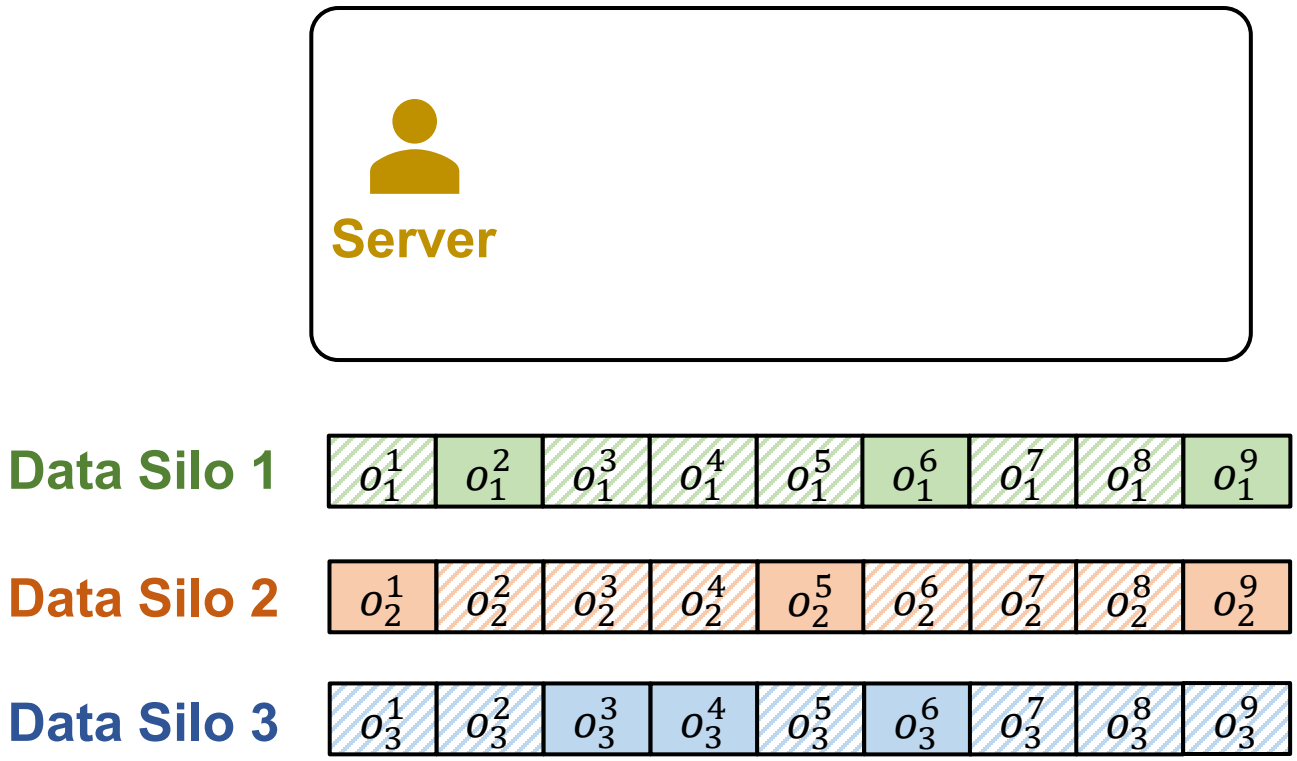
θ: a smoothing factor that ensures all silos maintain a non-zero selection probability

Contribution-based FANNS

- Basic Idea: select silos according to their cumulative contributions over previous rounds
  - Specifically, if silo S_i has contributed t_i objects to the current top-k results, its sampling probability is set proportional to $t_i + \theta$
  - The smoothing factor θ decays exponentially with factor τ ($\tau < 1$)
    - Early stage: unreliable contributions → larger θ to encourage exploration
    - Later stage: reliable contributions → smaller θ to exploit top silos

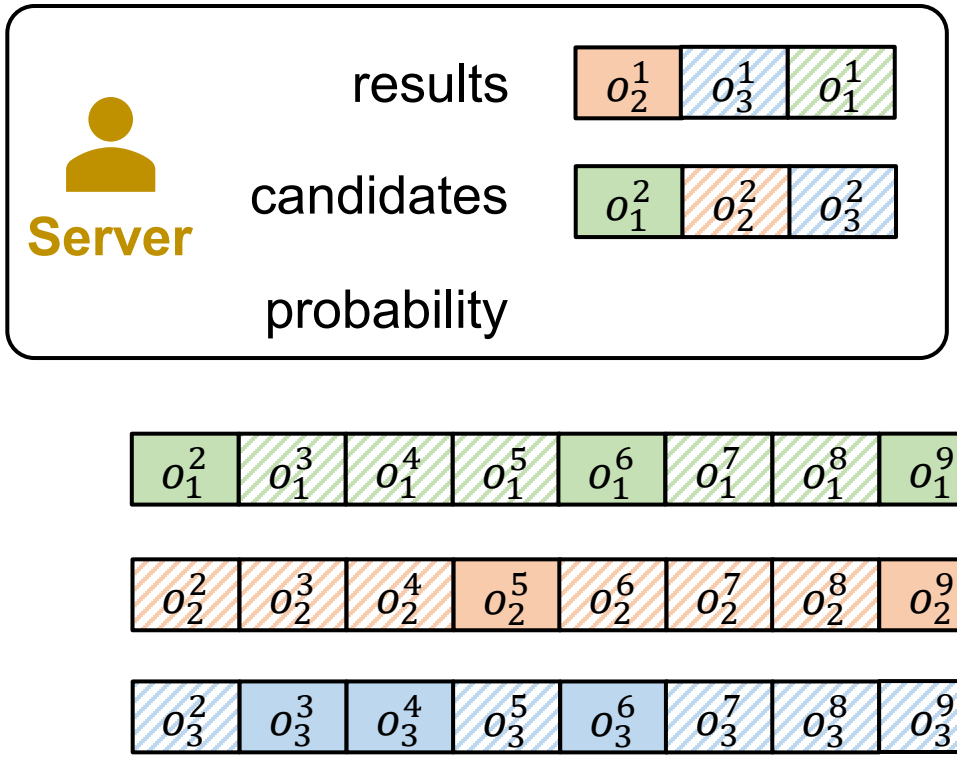
Contribution-based FANNS

- Running Example ($k = 3$)
  - The beginning resembles competition-based method



Contribution-based FANNS

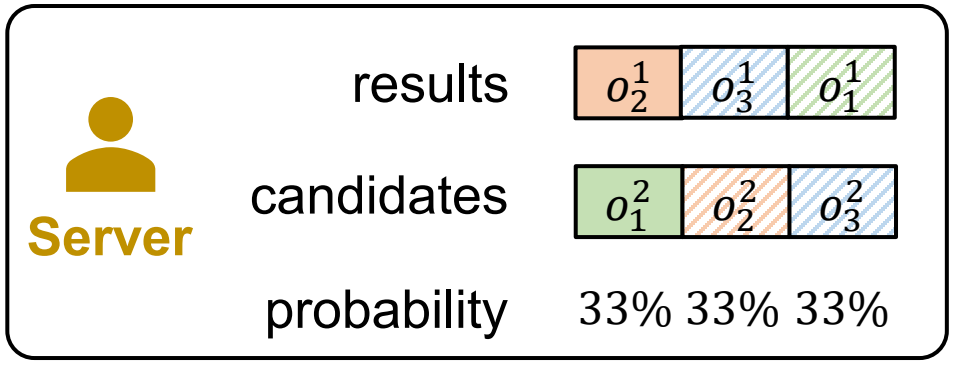
- Running Example ($k = 3$)
  - Select top candidate from each silo

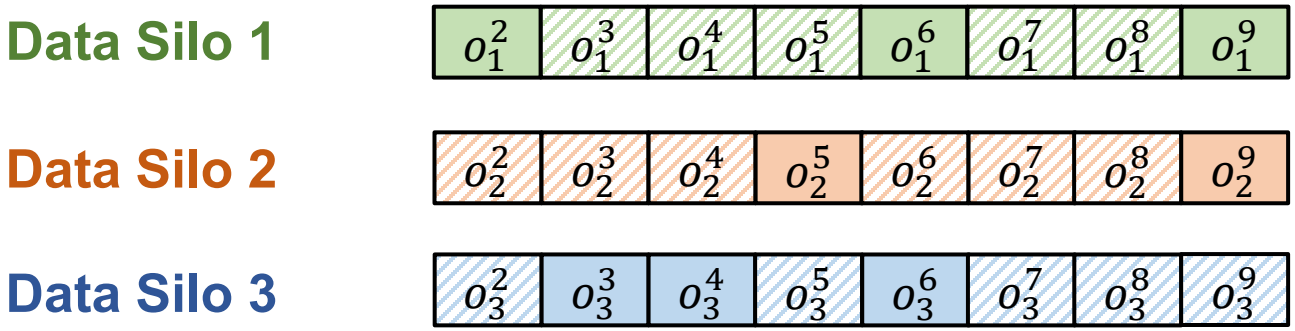


Contribution-based FANNS

- Running Example ($k = 3$)
  - In the beginning, $\theta = 1$
  - Compute probability of sampling:

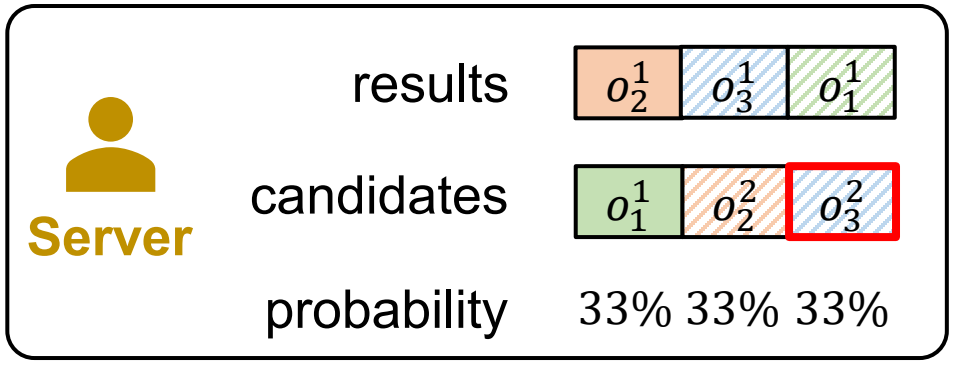
$$1 + 1 : 1 + 1 : 1 + 1 = 33\% : 33\% : 33\%$$

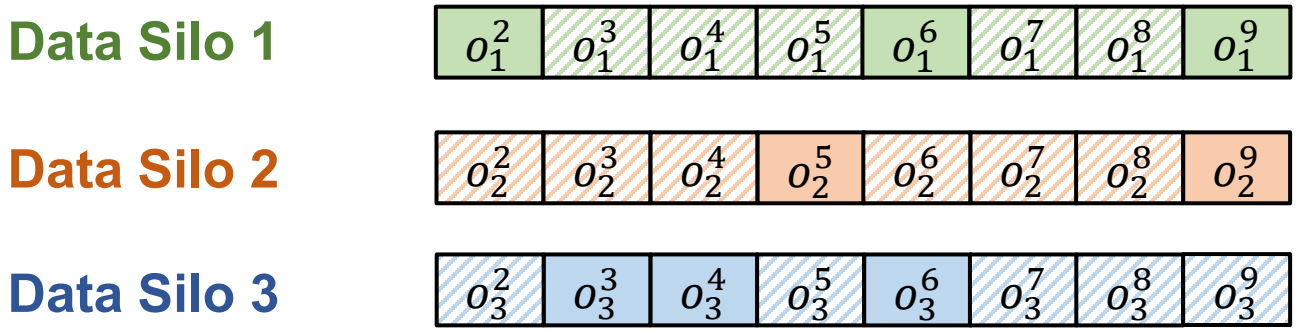




Contribution-based FANNS

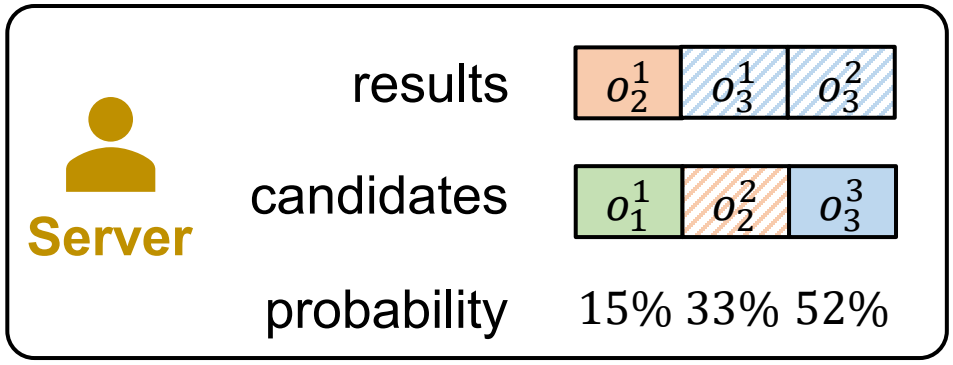
- Running Example ($k = 3$)
  - Sample silo 3, requiring it to provide the next candidate

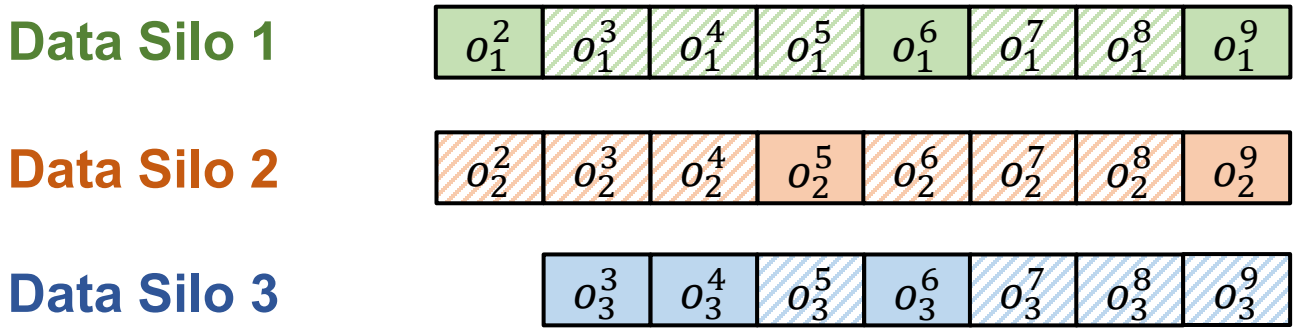




Contribution-based FANNS

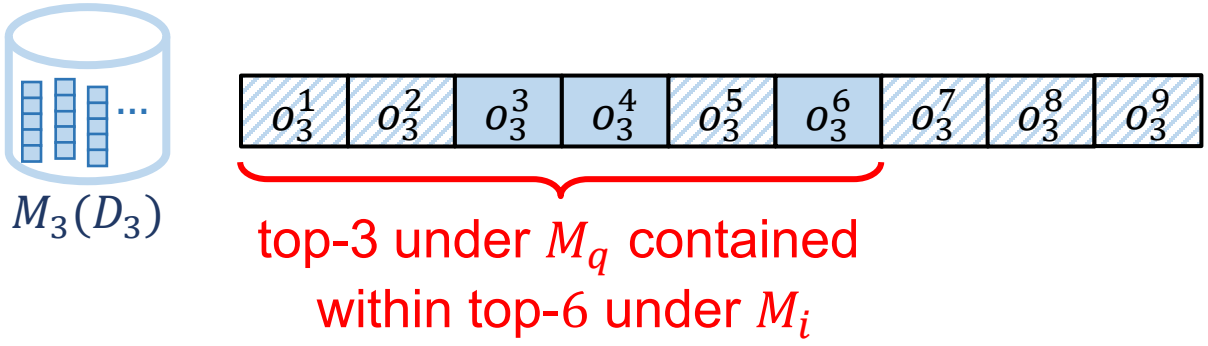
- Running Example ($k = 3$)
  - Smoothing factor decays to $\theta = 0.8$
  - The sampling probabilities have changed to
    $0 + 0.8 : 1 + 0.8 : 2 + 0.8 = 15\% : 33\% : 52\%$





Theoretical Analysis

- Measure for embedding heterogeneity
  - Define the deviation of M_i from M_q as $\delta_i = \frac{k'}{k}$, if k nearest neighbors of q under M_q are contained within the top k' nearest neighbors under M_i
  - For example, the deviation of M_3 is $\frac{6}{3} = 2$



  - Empirically, δ_i typically falls between 10 and 100 across datasets and models

Theoretical Analysis

- Considering a data silo S_i, which contributes k_i objects to the FANNS result, and the deviation of its local model is δ_i, then:
  - Expected number for re-embedding operations in competition-based algorithm is $O(\delta_i k_i n)$
  - Expected number for re-embedding operations in contribution-based algorithm is $O(\delta_i k \log(\frac{nk_i}{k} + 1))$

Theoretical Analysis

- For all the value of k_i and δ_i,

$$\delta_i k_i n \geq \delta_i k \log(\frac{nk_i}{k} + 1)$$

  - Indicating that contribution-based algorithm always requires fewer re-embeddings asymptotically

- Under skewed result distribution (common in federated environment) where $k_i \approx k$
  - Re-embedding operations in contribution-based algorithm is $O(\delta_i k_i \log n)$
  - Remarkable reduction over the $O(\delta_i k_i n)$ cost by the competition-based algorithm

Outline

- Background
- Problem Statement
- Method
- Experiment
- Conclusion

Experiment: Setup

- Datasets

Dataset	Data Type	Cardinality	Partition
Sentiment [1]	Text	100k	Natural
Reddit [1]	Text	837k	Natural
MS MARCO [2]	Text	1.6M	Manual
i-Naturalist [3]	Image	3.8M	Manual

Partitioning according to Dirichlet distribution, following prior work

[1] Leaf: A Benchmark for Federated Settings. *Arxiv 2018*.
[2] MS MARCO: A Human-Generated Machine Reading Comprehension Dataset. *Arxiv 2016*.
[3] The i-Naturalist Species Classification and Detection Dataset. *CVPR 2018*.

Experiment: Setup

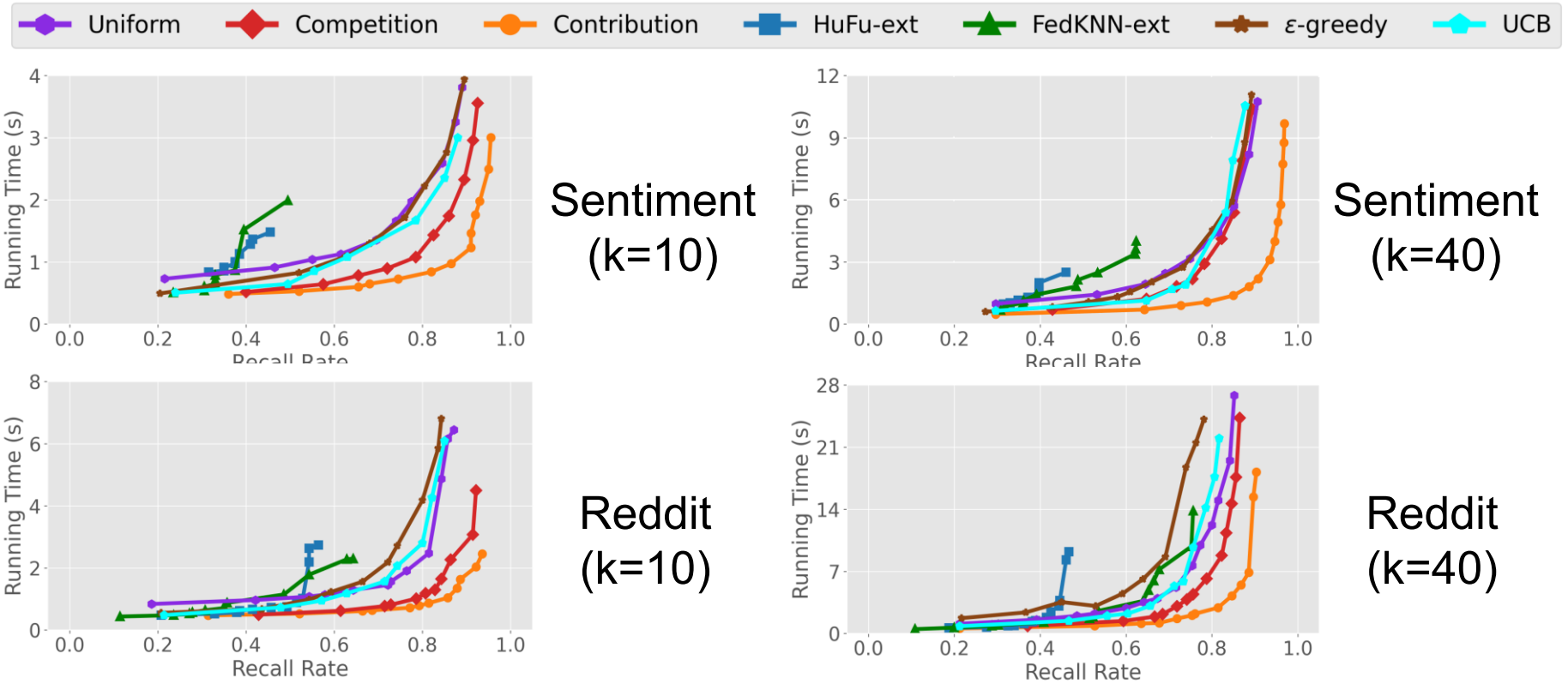
- Compared Method
  - HuFu-ext [1] : Spatial data federation system
  - FedKNN-ext [2] : Federated kNN algorithm
  - ε-Greedy [3] : Classic bandit algorithm
  - UCB [3] : Classic bandit algorithm

- Metrics for Search Performance
  - Recall rate: $\frac{|r \cap r^*|}{k}$ (r: returned result, r^*: ground truth)
  - Latency: query processing time
  - Communication cost: total amount of data transmitted

[1] Hu-Fu: Efficient and Secure Spatial Queries over Data Federation. *PVLDB 2022*.
[2] FedKNN: Secure Federated k-Nearest Neighbor Search. *SIGMOD 2024*.
[3] Bandit algorithms. *Cambridge University Press 2020*.

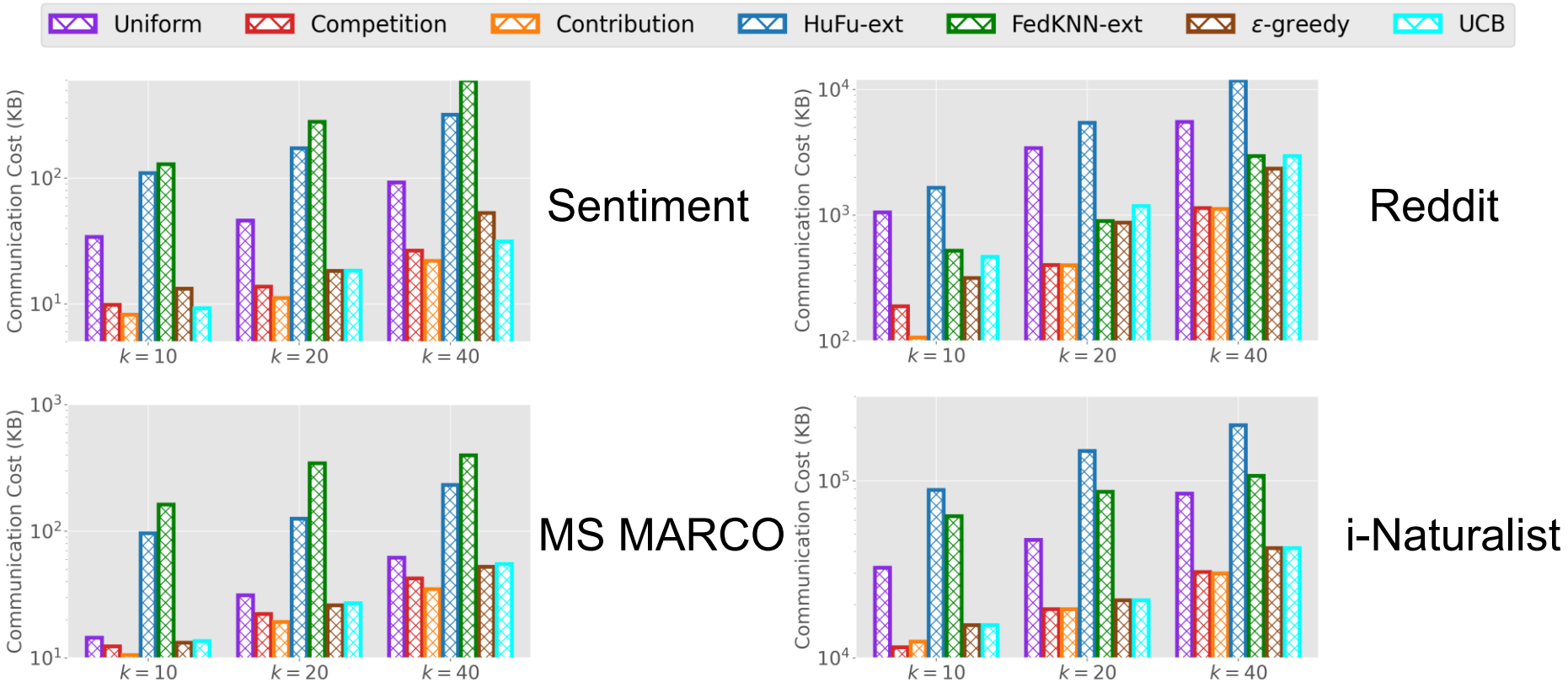
Experiment: Result

- End-to-end Performance
  - Yield recall rate of over 90% across datasets.
  - Improves the query efficiency by 2.3× to 6.2× over the baselines under the same accuracy.



Experiment: Result

- End-to-end Performance
  - Reduce the communication cost by 31.4% to 96.3% compared to the baselines



Outline

- Background
- Problem Statement
- Method
- Experiment
- Conclusion

Conclusion

- We introduce the problem of Federated Approximate Nearest Neighbor Search (FANNS) under **embedding heterogeneity**.

- We design two algorithms, **competition-based** and **contribution-based** FANNS, and provide theoretical guarantees on their re-embedding costs.

- Our solution yields a recall rate of over 90% across four datasets, and improves the query efficiency by 2.3× to 6.2× over the baselines.

Thank You!