Poster: Scalable Private Top-k Retrieval via Rank-Based Selection

Jae-yun Kim, Jieun Yun, Saerom Park
Ulsan National Institute of Science and Technology
Ulsan, Republic of Korea
{yistar, dbswldms316, srompark}@unist.ac.kr

***Abstract*—Privacy-preserving top-k retrieval is a fundamental primitive for encrypted search and machine learning inference, but remains computationally prohibitive under Fully Homomorphic Encryption due to the heavy cost of encrypted comparisons, especially for large-scale arrays. In this paper, we present a scalable and efficient privacy-preserving top-k retrieval algorithm based on the CKKS scheme to address this bottleneck. Our approach introduces a novel rank-based selection mechanism with shallow multiplicative depth and a multi-stage 'tournament-style' candidate refinement strategy with optimized parallelization. Through experiments, we demonstrate that our algorithm achieves unprecedented practical efficiency at scale, processing 131,072 encrypted entries in just 14-30 minutes for various k.**

I. Introduction

Fully Homomorphic Encryption (FHE) provides a powerful primitive for privacy-preserving computations on encrypted data, enabling a wide range of applications without compromising confidentiality [1]. Despite its promise, FHE operations often incur significant computational overhead and are limited to a restricted set of basic functionalities. While recent research has addressed some operational constraints, FHE-based ordering and selection algorithms still face scalability issues. Traditional sorting or ranking algorithms often rely on data-dependent operations (e.g., dynamic pivots in QuickSort) or iterative procedures that necessitate frequent and costly bootstrapping operations in FHE. Thus, current FHE-based solutions struggle to scale effectively, hindering their practical deployment on large data [2, 3].

In this study, we present a scalable privacy-preserving top-k retrieval algorithm using the CKKS scheme [1], achieving significant efficiency through SIMD and multi-threaded parallelizations. Our method introduces two core components: a shallow-depth rank-based selection and a multi-stage candidate refinement algorithm, effectively decouples comparison overhead from dataset size, addressing a key scalability bottleneck.

II. Background

In this study, we construct a private top-k retrieval algorithm with scalability, constructed on FHE. Our work uses the CKKS scheme [1], which supports approximate arithmetic over complex numbers and SIMD-style parallelism over N slots. Only arithmetic operations and slot permutations are supported directly, and combinations of such operations allow various linear transformations regarding the slots as matrices.

We consider a non-interactive two-party setting where a client holds private data and a semi-honest server performs computation. The client generates FHE keys, shares the evaluation key with the server, and sends encrypted data. The server, using only the evaluation key and ciphertexts, executes the top-k retrieval algorithm and returns the encrypted result. No interaction is required beyond the initial key and data transfer. Throughout the process, intermediate values remain hidden under the IND-CPA security of the CKKS scheme.

III. Private Rank-Based Selection Algorithm

We design the RankSelect algorithm as a core building block for our top-k retrieval method, enabling the extraction of elements with specified ranks from an input vector $\boldsymbol{a} = (a_0, \ldots, a_{n-1}) \in \mathbb{Z}_p^n$. The algorithm proceeds in three stages: (i) constructing a comparison matrix, (ii) computing the encrypted rank vector, and (iii) reordering the encrypted input according to the computed ranks.

Construct the comparison matrix: We construct a binary comparison matrix $\boldsymbol{B} \in \{0,1\}^{n \times n}$ with entries $b_{ij} = \mathbb{I}(a_i > a_j) \oplus \mathbb{I}(a_i = a_j \text{ and } i \geq j)$, where $\oplus$ denotes the logical XOR operation. The construction proceeds via the following: (i) generate two ciphertexts with the same dimensions as the comparison matrix by replicating the input vector along rows and columns, respectively; (ii) compute their difference; and (iii) apply the encrypted sign function from [2].

Compute the rank vector: The rank vector $\boldsymbol{w} \in \mathbb{Z}_n^n$ is computed from the row-wise sums of $\boldsymbol{B}$. Then, each entry of $\boldsymbol{w}$ represents the rank of the corresponding element in $\boldsymbol{a}$.

Reorder the input array: Using the encrypted indicator function from [4], we perform an equality test between the rank vector $\boldsymbol{w}$ and a target rank i, producing a mask vector that identifies the position of the i-th largest element. For improved efficiency, RankSelect exploits the SIMD capability of the CKKS scheme to extract multiple elements simultaneously. Thus, it can retrieve the top-k elements from n candidates in a single batched operation.

Note on Precision. The CKKS-based encrypted sign function [2] and indicator function [4] assume discrete-valued inputs, with precision controlled by multiplicative depth consumption. Our algorithms inherit this assumption, and correctness is guaranteed provided the input domain is discrete and sufficient depth is allocated.

IV. Private Top-k Retrieval

We propose a scalable private top-k retrieval method by introducing an efficient packing strategy and an optimized candidate reduction procedure. For scalability, our approach operates for large input arrays in multiple stages, progressively reducing the candidate set through repeated applications of RankSelect. Our algorithm optimizes the sub-block packing

size through the estimation of the overall parallel cost of RankSelect under hardware constraints on parallel processing.

Packing Strategy: Starting from an initial set of $n_0 = n$ candidates, we need to partition the input into sub-arrays of size u_0 when the comparison matrix can not be packed into a single ciphertext with slot size N. The maximum sub-array size is set to $u_0 = 2^{\lfloor \log \sqrt{N} \rfloor}$, as this minimizes the candidate reduction ratio k/u_0, thereby reducing the total number of rounds R_M. A larger u_0 increases the number of ciphertexts, given by $T_0 = \frac{n_0 u_0}{N}$, which is advantageous in a parallel computation setting. However, when parallel resources are limited, we select the smallest feasible u_0 that maintains R_M, thereby reducing the number of ciphertexts.

Multi-ciphertext Stage: When dealing with large inputs where the candidates are distributed across multiple ciphertexts ($T_i > 1$), we apply RankSelect in parallel within each ciphertext. After applying RankSelect at round i, the number of candidates n_i is reduced to $n_{i+1} = n_i \cdot k/u_i$. For the reduced set of candidates, we design a light merge algorithm that consolidates multiple ciphertexts into a single output using additions and rotations. The group size L for merge is set to u_i/k, which reduces the number of ciphertexts to $T_{i+1} = T_i/L$. The maximum sub-array size for the next round is set to $u_{i+1} = \min(u_i^2/k, 2^{\lfloor \log \sqrt{N} \rfloor})$, where $N/(n_i k/u_i) = u_i^2/k$. This process continues until the number of candidates, n_i, is small enough to be consolidated into a single ciphertext, transitioning to the next stage.

Single-ciphertext Stage: When all candidates can be packed into a single ciphertext $T_i = 1$, we apply RankSelect in parallel within a ciphertext to reduce the number of top-k candidates from n_i to $n_{i+1} = k \cdot n_i^2/N$. This iterative reduction continues while $n_i > \sqrt{N}$.

Final top-k Retrieval Stage: Once the number of candidates is sufficiently small ($n_i \leq \sqrt{N}$), a final RankSelect operation is performed to extract the top-k elements.

Thus, our approach features the following advantages:

- **Space Efficiency**: Our partitioning approach drastically reduces the number of ciphertexts required compared to the comparison matrix-based approach [3], which suffers quadratic expansion in slot requirements.
- **Adaptive Parameter Tuning**: We adaptively update the sub-array size u_i using a carefully designed ciphertext merge criterion, ensuring fast candidate reduction and minimizing round count, which consequently reduces overall cost, especially bootstrapping overhead.
- **Enhanced Efficiency via Parallelization**: Through multi-threaded parallelization, we demonstrate the practicality and scalability of our algorithm.

V. Experiments

Our experiments were conducted on a server with an Intel Xeon 6426Y at 2.5GHz and 384GB RAM, with a maximum of 128 threads. Our CKKS implementation is based on the OpenFHE library [1], using a polynomial ring of degree 2^{17} (providing $N = 2^{16}$ slots) with 128-bit security parameters. All reported runtimes are averaged over 8 independent iterations.

Private Top-1 Retrieval. For the top-1 retrieval, PrivTopk achieved runtimes ranging from 371.37s ($n = 2^9$) to 830.77s ($n = 2^{17}$). The primary advantage stems from its ability to perform faster candidate reduction per round through large SIMD comparisons, outperforming both scalable bitonic sorting (**Engorgio**) [5] already requiring 3322.82s at $n = 2^9$ and tournament-based methods (**NEXUS**) [6] reporting 1281.60s ($n = 2^9$) to 2565.65s ($n = 2^{17}$). The matrix-comparison approach (**MEHP**) [3] exhibits a more rapid cost increase when processing multiple ciphertexts since the required number of ciphertexts grows quadratically, reaching 2^{12} ciphertexts and achieving 4037.18s at $n = 2^{14}$, beyond which it exceeds 2 hours. Our design maintains low round complexity and offers superior scalability compared to them.

Private Top-k Retrieval. PrivTopk exhibits only a gradual increase in the number of rounds and requires no more than 8 bootstrapping operations across all experiments, achieving 1775.03s even at the largest scale ($k = 16, n = 2^{17}$). In contrast, **Engorgio** [5] incurs substantial multiplicative depth and thus suffers from severe performance degradation, requiring 6540.86s even at the smallest scale ($k = 2, n = 2^9$).

[1] https://openfhe.org/

Acknowledgments

This work was supported by IITP grant funded by the Korea government (MSIT) (No. RS-2024-00398360, 60%) and ITRC grant funded by the Korea government (MSIT)(IITP-2025-RS-2024-00436936, 20%). This work was also supported by the NRF of Korea grant funded by the Korea government (MSIT) (RS-2025-00515481, 20%).

References

[1] J. H. Cheon, A. Kim, M. Kim, and Y. S. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *ASIACRYPT (1)*, vol. 10624. Springer, 2017, pp. 409–437.

[2] J. H. Cheon, D. Kim, and D. Kim, "Efficient homomorphic comparison methods with optimal complexity," in *ASIACRYPT (2)*, ser. Lecture Notes in Computer Science, vol. 12492. Springer, 2020, pp. 221–256.

[3] F. Mazzone, M. Everts, F. Hahn, and A. Peter, "Efficient ranking, order statistics, and sorting under ckks," in *34th USENIX Security Symposium*. USENIX Association, aug 2025.

[4] J. Kim, S. Park, J. Lee, and J. H. Cheon, "Privacy-preserving embedding via look-up table evaluation with fully homomorphic encryption," in *Forty-first International Conference on Machine Learning*, 2024.

[5] S. Bian, H. Pan, J. Hu, Z. Zhang, Y. Fu, J. Hua, Y. Chen, B. Zhang, Y. Jin, J. Dong, and Z. Guan, "Engorgio: An arbitrary-precision unbounded-size hybrid encrypted database via quantized fully homomorphic encryption," *IACR Cryptol. ePrint Arch.*, p. 198, 2025.

[6] J. Zhang, J. Liu, X. Yang, Y. Wang, K. Chen, X. Hou, K. Ren, and X. Yang, "Secure transformer inference made non-interactive," *IACR Cryptol. ePrint Arch.*, p. 136, 2024.

Scalable Private Top-k Retrieval via Rank-Based Selection

Jae-yun Kim, Jieun Yun, Saerom Park
yistar@unist.ac.kr, dbswldms316@unist.ac.kr, srompark@unist.ac.kr
Ulsan National Institute of Science and Technology, Ulsan, Republic of Korea

Motivation

Fully Homomorphic Encryption (FHE) provides a powerful primitive for privacy-preserving computations on encrypted data.

However, FHE-based ordering and selection algorithms still face **scalability issues** while sorting/top-k are fundamental primitive for encrypted search and machine learning inference.

Key Challenges

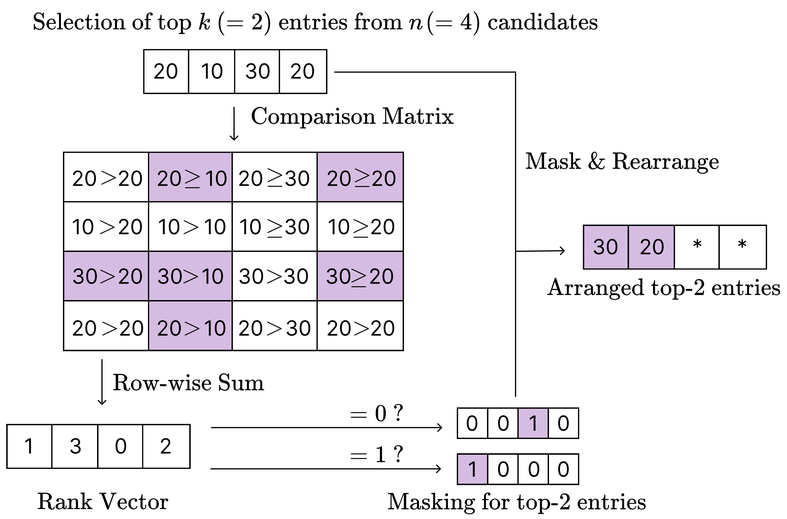
- FHE operations are often limited to a **restricted set of basic functionalities** as arithmetic operations and slot permutations.
- Requires shallow-depth circuit to avoid **heavy cost of bootstrapping**, which resets multiplicative depth consumed.
- Shallow circuit often requires **quadratically increasing number** of comparisons & ciphertext counts [1], resulting weak scalability.

Problem Setting & Goal

We present *a scalable encrypted top-k algorithm* on **CKKS scheme**, applicable to **non-interactive two-party scenario**.
This approach effectively decouples comparison overhead from dataset size, addressing a key scalability bottleneck.

Private Rank-Based Selection

RankSelect: For input ciphertext with n entries with CKKS slot size N, it simultaneously selects the elements of specific rank leverage *comparison-matrtix based approach*.

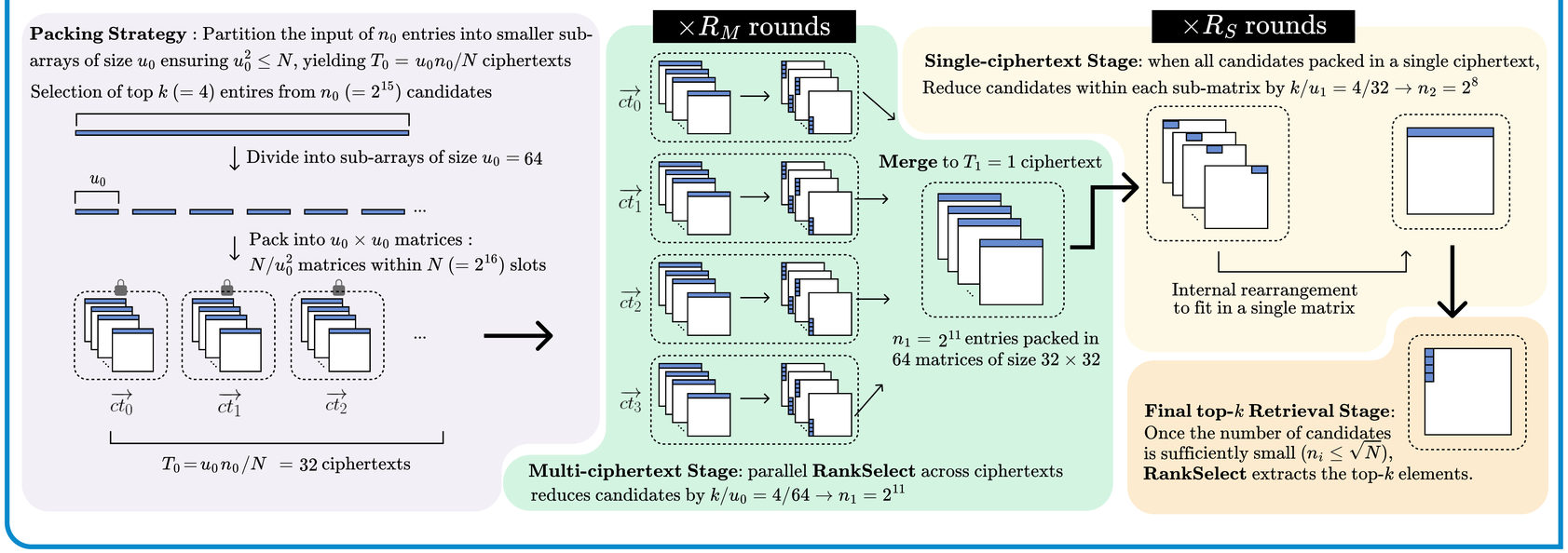


- Leverages an **improved tie-breaking method** and adopt **efficient equality test algorithm**, compared to the previous work [1].
- Applicable to both sort and top-k, since it can simultaenously select entries of specific ranks.
- Efficiency is maximized when $n \leq \sqrt{N}$.
- **Only two encrypted comparison** required to sort $\sqrt{N}$ elements.

Private Top-k Retrieval

We propose **PrivTopk**: a scalable private top-k retrieval method through multiple stages of candidate reduction, via repeated applications of **RankSelect** under hardware constraints on parallel processing.

- **Efficient Packing for Parallelism**: We choose a larger initial block size u_0 to reduce the number of rounds (R_M, R_S), at the cost of increasing the initial ciphertext count $T_0 = u_0 n_0 / N$, which may limit efficiency under constrained parallel resources.
- **Notation**: At round i, n_i, u_i, and T_i denote the number of candidates, sub-array size, and number of ciphertexts, respectively.
- **Example Workflow**: Our method selects the top $k = 4$ elements from $n_0 = 2^{15}$ candidates with $R_M = 1$ and $R_S = 1$.

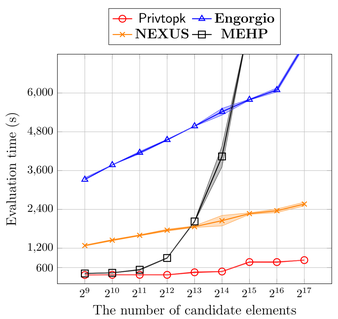


Primary Results

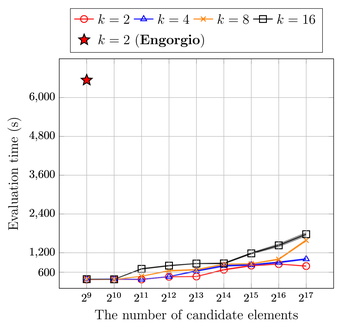
Baselines

- **MEHP** [1]: a comparison matrix-based sorting algorithm
- **NEXUS** [2]: a binary tournament-style argmax algorithm
- **Engorgio** [3]: an encrypted version of bitonic sorting algorithm

Experimental Results on 128-thread environment for $2^9 \leq n \leq 2^{17}$



Evaluation time comparison for private top-1 retrieval.



Evaluation time comparison for various k ($2 \leq k \leq 16$).

The results demonstrate:

- **Sublinear cost growth** due to logarithmic round complexity, vs. quadratic cost growth of **MEHP** at large scales.
- **Accelerated reduction rate per round** outpacing binary tournament (**NEXUS**) and bitonic sorting (**Engorgio**).

Advantages of Our Approach

Space Efficiency: Our partitioning approach *drastically reduces the number of ciphertexts required* compared to the comparison matrix-based approach (**MEHP**), which suffers quadratic expansion in slot requirements.
Adaptive Sub-array Size Update: We adaptively update the sub-array size u_i using a carefully designed ciphertext merge criterion, *ensuring fast candidate reduction and minimizing round count*, consequently reducing overall cost, especially bootstrapping overhead.
Enhanced Efficiency via Parallelization: Through multi-threaded parallelization, we demonstrate *the practicality and scalability* of our algorithm.

As a result, our algorithm achieved shallow depth top-k algorithm with the **state-of-the-art scalability** (2^{17} entires).

References

[1] F. Mazzone *et al.*, "Efficient ranking, order statistics, and sorting under ckks," in *34th USENIX Security Symposium (USENIX Security '25)*, 2025.

[2] J. Zhang *et al.*, "Secure transformer inference made non-interactive," *IACR Cryptol. ePrint Arch.*, 2024.

[3] S. Bian *et al.*, "Engorgio: An arbitrary-precision unbounded-size hybrid encrypted database via quantized fully homomorphic encryption," *IACR Cryptol. ePrint Arch.*, 2025.