Journal of XYZ, 20xx, 1–22

doi: xx.xxxx/xxxx
Manuscript in Preparation
Paper

PAPER

FEDRANN: effective overlap graph construction based on dimensionality reduction and approximate nearest neighbors

Jia-Yuan Zhang[1, 2, 3, 4, †], Changjiu Miao[4, †], Teng Qiu[3], Xiaoshuang Xia[5], Lei He[5], Junyi He[4], Yuhui Sun[6], Tao Zeng[6], Yuxiang Li[5], Xun Xu[6,7*], Yijun Ruan[1,*] and Yuliang Dong[3,8*]

[1]Life Sciences Institute, Zhejiang University, Hangzhou, Zhejiang 310058, China and [2]State Key Laboratory of Genome and Multi-omics Technologies, BGI Research, Hangzhou 310030, China and [3]BGI Research, Hangzhou 310030, China and [4]BGI Hangzhou CycloneSEQ Technology Co., Ltd, Hangzhou 310030 , China and [5]BGI Research, Wuhan 430074, China and [6]State Key Laboratory of Genome and Multi-omics Technologies, BGI Research, Shenzhen 518083, China and [7]Guangdong Provincial Key Laboratory of Genome Read and Write, BGI Research, Shenzhen 518083, China and [8]BGI Research, Shenzhen 518083, China

*Correspondence: Yuliang Dong (dongyuliang@genomics.cn), Yijun Ruan (yjruan@zju.edu.cn), Xun Xu (xuxun@genomics.cn)
†These authors contributed equally.

Abstract

De novo genome assembly reconstructs genomes from sequencing reads without a reference, a critical process enhanced by long-read sequencing technologies. Most current assembly methods use the overlap-layout-consensus (OLC) paradigm, where overlap graph construction remains a computational bottleneck due to large data volumes and repetitive genomic sequences. Traditional overlap detection relies on seed-and-extension or locality-sensitive hashing (LSH) strategies such as MinHash, each with trade-offs in accuracy, efficiency, and sensitivity to sequence properties. Limitations of these approaches include sensitivity to repetitive k-mers and bias against longer sequences. To address these challenges, we propose a novel overlap graph construction workflow based on three steps: feature extraction, dimensionality reduction, and k-nearest neighbors (k-NN) search. Benchmarking on real long-read datasets demonstrates high accuracy, versatility and scalability of this approach, offering a promising alternative for efficient and accurate overlap graph construction in genome assembly.

Key words:
de novo assembly, overlap-layout-consensus, locality-sensitive hashing, dimensionality reduction, k-nearest neighbors

Introduction

De novo assembly reconstructs a genome from scratch using overlapping sequencing reads without a reference, which is crucial for studying novel genomes, uncovering structural variations, and exploring genetic diversity [1]. The advancement of long-read sequencing technologies, such as PacBio, Oxford Nanopore Technologies and BGI CycloneSEQ [2] platforms, has significantly improved *de novo* assembly by generating reads spanning repetitive regions and complex structural variants [3, 4]. Most long-read *de novo* assembly methods rely on the Overlap-Layout-Consensus (OLC) approach, which aligns reads based on overlaps and iteratively refines contigs to reconstruct high-quality genomes [5, 6, 7]. A key step in the OLC approach is overlap graph construction,

Compiled on: June 10, 2025.
Draft manuscript prepared by the author.

which involves finding overlaps from a large collection of sequencing reads, where the existence of an overlap between a pair of reads is typically determined by their sequence similarity. The construction of overlap graphs faces two primary challenges: the computational burden of processing large amount of sequences (millions of reads for human-sized genomes), which demands efficient algorithms and substantial memory resources, and the inherent complexity caused by repetitive genomic sequences that create ambiguous overlaps [8]. These repeats often lead to fragmented or erroneous connections in the graph, significantly complicating the assembly process and potentially introducing misassemblies. This issue is further exacerbated by sequencing errors, which introduce spurious overlaps and masking true overlaps. Overcoming these obstacles is crucial for accurate genome reconstruction.

Existing methods for overlap graph construction can be roughly categorized into two classes: seed-and-extension and locality-sensitive hashing (LSH) methods. seed-and-extension approaches, exemplified by alignment tools like BLAST [9], DALIGN [10], and Minimap2 [11], first identify small exact or near-exact matches (seeds) between the query sequence and the target sequences. Seed matches are then filtered based on local density or other criteria. For each retained seed, the algorithm attempts to extend the alignment in both directions using dynamic programming (DP) to build a more comprehensive alignment. Finally, target sequences with the best alignments (according to certain heuristic criteria) are identified as overlapping sequences. As seed-and-extension methods only perform time-consuming DP-based alignment on high-potential regions that contain many seed matches, they scale well to large databases and long query sequences. However, the seed matching and filtering steps are heuristic in their nature, and are prone to returning target sequences that are locally but not globally similar to the query sequence (false positives) [12]. In addition, these methods often depend on many parameters such as minimum seed density, which require careful tuning to achieve a good trade-off between accuracy and computational efficiency. As a result, overlap graph construction methods that depend on the seed-and-extension strategy, such as xRead [13] and the self-correction modules of NECAT [14] and PECAT [7], typically have many parameters that need to be fine-tuned for each sequencing platform based on its error profile.

LSH-based methods, in particular MinHash, have also been widely used in overlap graph construction [15, 5]. MinHash was originally developed for text mining tasks, e.g. finding similar news articles [16]. In MinHash, each sequence is first encoded as an unordered set of unique tokens, which are typically k-mers (subsequences of fixed-length k). The MinHash algorithm then applies multiple independent hash functions and records the minimum hash value of each set under each hash function. The resulting "sketch" serves as a compact approximation of the full set. Overlapping sequences will have more similar sketches compared to unrelated sequences. Popular *de novo assembly* tools that implement variants of MinHash include MHAP [15], Canu [5], and Shasta [6]. In addition, minimizers, a seed matching scheme based on the same core ideas of MinHash, is used in popular seed-and-extension methods including Minimap2 [11].

Despite the successful application of MinHash in bioinformatics, it has two major limitations. First, all tokens are treated equally in MinHash. In practice, highly repetitive tokens, such as k-mers from low-complexity genomic regions, can dominate the similarity measure, overshadowing more informative, unique tokens [5]. Second, the MinHash algorithm effectively estimates the Jaccard index between two sets, which represents the proportion of shared tokens between the two sequences they encode [16, 17]. When the two sequences vary greatly in lengths, the measuring their similarity using the Jaccard index can be ineffective [6, 18].

To address the limitations of existing methods for overlap graph construction, we developed a new workflow inspired by common single-cell sequencing workflows. Our approach, hereby referred to as FEDRANN, comprises three main steps: feature extraction (FE), dimensionality reduction (DR), and approximate nearest neighbor (ANN) search. We first designed a benchmarking pipeline to evaluate overlap graph accuracy using both simulated and real long-read sequencing data. Using this pipeline, we systematically compared various combinations of feature extraction, DR, and ANN methods across datasets of differing sizes and complexities. We then benchmarked the best-performing method, termed ISN, against established tools such as Minimap2, MHAP, MECAT2, wtdbg2, xRead and BLEND. Our results show that ISN achieves higher accuracy in overlap graph construction while substantially reducing runtime, highlighting its potential utility in *de novo* genome assembly.

Results

Overview of the FEDRANN workflow

Many text-mining algorithms, such as MinHash and SimHash [19], employ the Bag-of-Words (BoW) model, which discards the ordering of tokens within a sequence and focuses solely on the presence or absence of each token, effectively encoding the sequence database as a high-dimensional sparse matrix [20]. We observed that, once sequences are encoded in this manner, constructing an overlap graph becomes analogous to performing a nearest-neighbor search in single-cell sequencing data, where both tasks rely on identifying similar items based on some similarity or distance metric in a high-dimensional sparse space (i.e., the sequence $\times$ token matrix or the cell $\times$ gene matrix). Based on this analogy, we hypothesized that dimensionality reduction and approximate nearest-neighbor (ANN) methods, which are widely used in single-cell data analysis, could also be applied to overlap graph construction with appropriate adaptations. Accordingly, we developed a new approach for overlap graph construction (Figure 1) consisting of three main steps: feature extraction, dimensionality reduction (DR), and ANN search, as described below.

Feature extraction. Given a dataset of n sequences, we uniformly sampled an alphabet of m k-mers and encoded the dataset as a high-dimensional matrix X with n rows and m columns, where each entry $X[i, j]$ represents the weight of k-mer j in sequence i. By default, the weights were binary, indicating the presence or absence of a k-mer. Optionally, term frequency (TF) and/or inverse document frequency (IDF) transformations could be applied to adjust the weight of each k-mer based on its frequency across the dataset. Since each sequence typically contains only a small subset of all possible k-mers, the resulting matrix is sparse.

Dimensionality reduction. Although it is possible to directly apply nearest-neighbor algorithms to the high-dimensional feature matrix, doing so is often computationally expensive due to the large number of dimensions. To mitigate this, we applied dimensionality reduction techniques to project each high-dimensional feature vector into a lower-dimensional space, while aiming to preserve the relative distances between sequences.

ANN search. Since exact k-nearest neighbor (k-NN) search has quadratic time complexity, it becomes infeasible for large genomic sequencing datasets with millions of reads. To address this, we employed approximate k-NN algorithms to identify the most similar sequences for each query sequence in the dataset. These candidate overlaps were then used to define edges in the resulting overlap graph.

Unlike seed-and-extension or LSH-based approaches that aim to identify all pairs of overlapping reads in a dataset, FEDRANN employs a k-nearest neighbors (k-NN) search to retrieve only the top k most similar reads for each query read. These nearest neighbors are expected to exhibit the longest overlaps with the query.

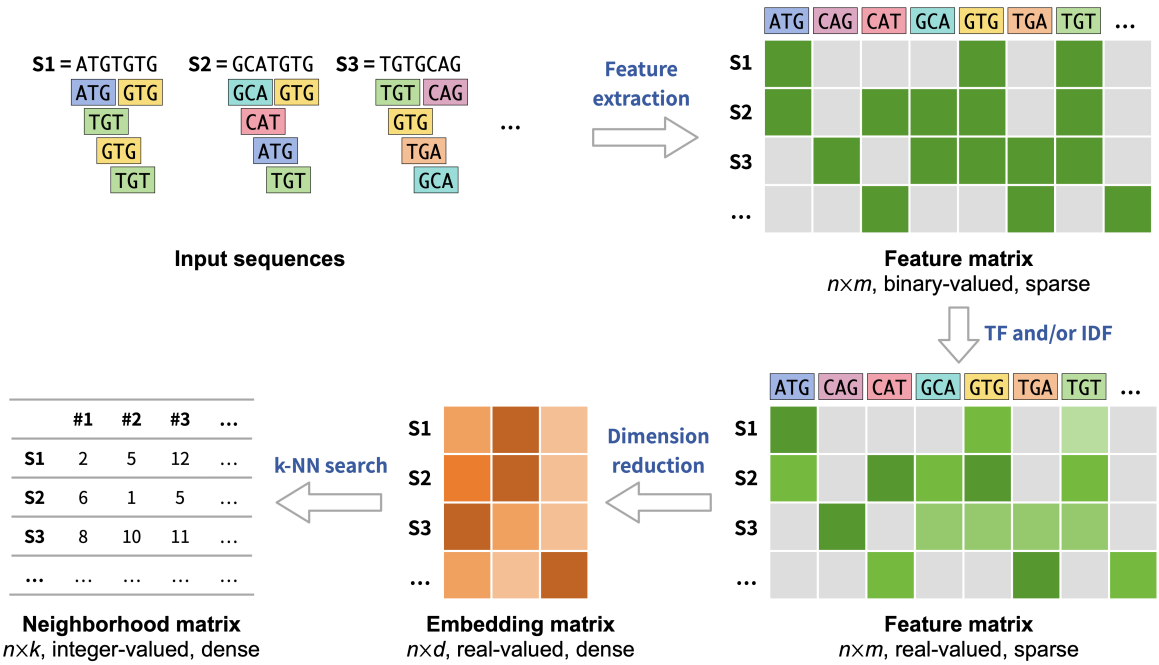

Figure 1. Diagrammatic representation of the proposed algorithm for sequence similarity search.
The algorithm is comprised of three key steps: (1) encoding the input sequences into a high-dimensional sparse feature matrix, with optional TF and/or IDF weighting; (3) applying a dimensionality reduction technique to embed the sequences into a space of lower dimensionality; (4) identifying sequence similarity through the use of a k-nearest neighbor search algorithm.

This approach aligns with the concept of the *best overlap graph* (BOG) [21, 5], which is grounded in the observation that full enumeration of overlapping read pairs is unnecessary, as shorter overlaps are typically redundant and pruned during graph refinement. For instance, the Shasta assembler retains by default only the top six edges per vertex in its overlap graph [6]. Restricting the number of edges in the overlap graph is advantageous, as it significantly reduces the complexity of subsequent graph layout computations.

Experimental setup

To assess specific overlap graph construction methods, we constructed overlap graphs for several long-read sequencing datasets and compared them against reference graphs generated using corresponding reference genomes (Figure 2). The datasets included Oxford Nanopore Technologies (ONT), PacBio HiFi and Cyclone-SEQ reads derived from various genomic regions or chromosomes of human, nematode, and fruitfly (Table S1). The accuracy and connectivity of each overlap graph construction method was evaluated using four metrics: precision, mean overlap size, singleton percentage and connected components number (see Methods for details).

k-NN outperforms Minimap2 and MinHash in repeat-rich sequences

We conducted a series of experiments to identify the most effective combinations of feature extraction, dimensionality reduction, and ANN methods for overlap graph construction. As a first step, we evaluated the impact of different weighting schemes (TF-IDF, TF, IDF and raw TF) during feature extraction, and compared similarity metrics (Euclidean distance and cosine distance) used in k-NN search. To isolate the effects of these factors, we computed exact k-nearest neighbors (ENN) without applying dimensionality reduction or ANN approximations.

Our results showed that using cosine distance for ENN search resulted in significantly higher precision and mean overlap size compared to Euclidean distance (Figures S1, S2, and S3). This is likely because Euclidean distance is sensitive to vector magnitude, while cosine distance captures only the angular difference between vectors, making it more robust to variations in sequence length. As a result, overlapping sequences of different lengths may still exhibit low cosine distances but high Euclidean distances, leading to better overlap detection when cosine similarity is used.

We evaluated the impact of different weighting schemes (TF, IDF, IDF and raw TF) on overlap graph quality. For baseline comparison, we also constructed a binary occurrence matrix that only records the presence (1) or absence (0) of k-mers in each read, discarding all frequency and weighting information. While TF-IDF has been used in assemblers like Canu [5] to help resolve repetitive sequences, our experiments indicated that IDF alone, rather than TF or TF-IDF, was more effective in improving graph quality. When cosine distance was used, applying IDF weighting led to better performance than both TF-IDF and unweighted representations, whereas TF alone degraded performance relative to the unweighted baseline (Figures S1, S2, and S3). These findings suggest that high-frequency k-mers—such as those originating from transposable elements, short tandem repeats, or other repetitive regions—are a major source of error in overlap graph construction. By down-weighting these features, IDF enhances the ability to distinguish between true overlaps. In contrast, TF and raw TF emphasizes frequent features and may counteract the benefits of IDF, thereby reducing accuracy.

We further compared the performance of overlap graphs constructed using IDF weighting and ENN based on cosine similarity against graphs generated by Minimap2 and MinHash, using the IGK region dataset (HC-2) as a benchmark. We evaluated edge precision and mean overlap length across different values of k, the number of neighbors used to construct the graph (Figure 3). As k increased, both precision and overlap length declined, reflecting the increased difficulty of retrieving accurate overlaps at larger neighborhood sizes. Nevertheless, the ENN+IDF approach consistently produced graphs with fewer erroneous edges and higher precision than the other methods, demonstrating its effectiveness

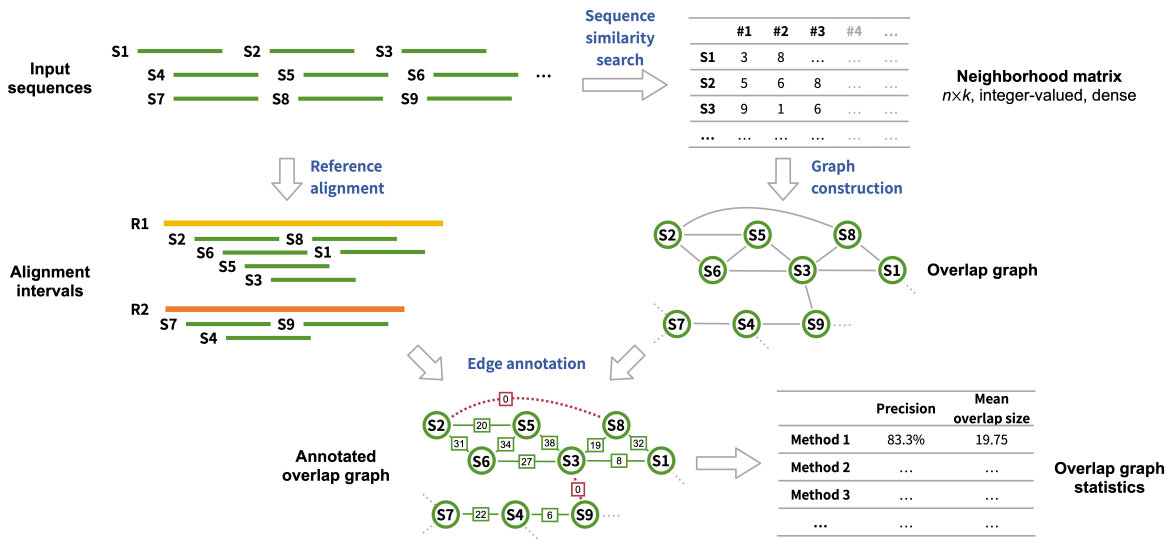

Figure 2. Assessment of sequence similarity search algorithms.
For the analysis of each algorithm, an overlap graph is constructed by linking each query sequence to the top k sequences identified as most similar by the algorithm. This overlap graph is subsequently assessed against a reference graph, which is obtained by aligning the input sequences to a reference genome to ascertain the accuracy and overlap size of each edge. The algorithm's performance is evaluated based on precision and mean overlap size.

even in repeat-rich regions such as the HLA.

Similar results were observed in other datasets, where the ENN approach yielded precision and overlap size comparable to or better than those achieved by MinHash and Minimap2 (Figures S4 and S5). These findings reinforce the potential of k-NN–based methods as a viable and accurate strategy for overlap graph construction.

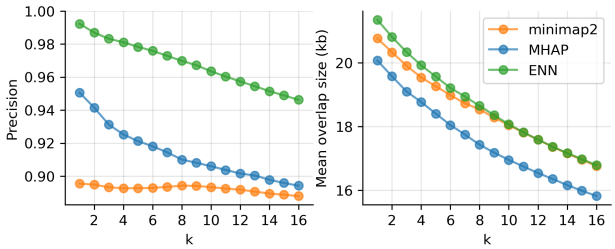

Figure 3. Accurate overlap identification using k-nearest neighbors.
Precision (left) and mean overlap size (right) of top k overlaps identified by seed-and-extension, MinHash and exact k-nearest neighbors method for dataset H1C.

Impact of sequencing characteristicson on k-NN performance

We systematically evaluated the impact of sequencing parameters on algorithm accuracy by generating simulated datasets with PBSIM3 for rice chromosome 1 (low-repeat but large region) and the human IGK locus (high-repeat region), covering read lengths of 10~30 kb, accuracy of 91%~99%, and depths of 10×~50×. The results confirmed that longer reads and lower error rates improved algorithm performance by providing more abundant and accurate sequence information. Notably, parameter sensitivity differed substantially between genomic contexts. In rice chromosome 1 (Chr1), which is a larger genomic region with relatively low repetitive content, sequencing read length emerged as the dominant factor influencing graph precision. Conversely, in the IGK regions – characterized by highly repetitive sequences that typically reduce graph precision – both evaluated parameters demonstrated balanced sensitivity.

Depth increase universally boosted performance, probably through two mechanisms: expanded k-mer sampling enabled robust statistical inference, and uniform coverage minimized neighbors misses. Crucially, high depth could compensate for suboptimal conditions. Benchmarking results on rice Chr1 demonstrated that 30 kb reads with 99% base accuracy achieved optimal precision at 10× sequencing depth. Notably, when increasing the depth to 20×, comparable precision could be attained using shorter 10 kb reads with 95% accuracy. This clearly illustrates the compensatory effect of sequencing depth: higher coverage can effectively offset limitations in read length and base-level accuracy when employing our algorithm.

Sparse random projection enables scalable and accurate dimensionality reduction

Dimensionality reduction (DR) techniques are widely used in single-cell sequencing analyses to reduce noise, accelerate computation, and facilitate data visualization. To evaluate their utility in overlap graph construction, we tested a range of DR methods, including linear approaches such as principal component analysis (PCA) [22] and sparse random projection (SRP), as well as non-linear methods including Uniform Manifold Approximation and Projection (UMAP) [23], Spectral Embedding, and scBimapping. We also evaluated SimHash [24], a locality-sensitive hashing (LSH)–based method related to random projection, which has been widely used in text mining and recently applied to biological sequence analysis [25, 26].

For each method, we first constructed the input feature matrix using IDF weighting, then applied the DR method, and finally computed exact k-nearest neighbors (ENN) from the resulting low-dimensional embeddings (Figure 1). We found that all tested DR methods except UMAP were able to generate accurate overlap graphs for smaller datasets (Figure 4a), suggesting that they preserved the relative pairwise distances between sequences sufficiently well. However, when applied to larger datasets (H5–H8), most methods failed to complete within the predefined resource constraints (wall clock time $\leq$ 6 hours; peak memory $\leq$ 1 TB) or failed due to matrix size exceeds limit. The only method that successfully scaled to the largest datasets, H3C, H3O and H3P, was SRP, GRP and SimHash. Time and memory profiling on dataset H3C fur-

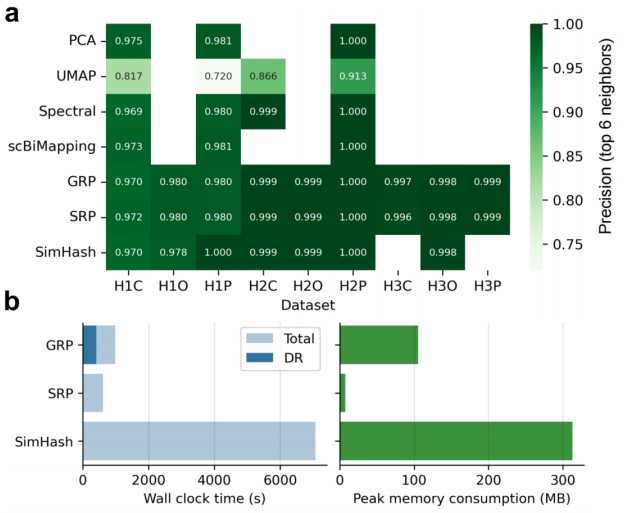

Figure 4. Evaluation of dimensionality reduction methods.
(a) Heatmap showing the overlap graph precision of various dimensionality reduction methods. Darker colors indicate higher precision at k = 6. Empty cells indicate that the dimensionality reduction method failed to finish under preset computational constraints (wall clock time $\leq$ 6 h; peak memory $\leq$ 1 Tb) or failed due to matrix size exceeds limit. **(b)** Wall clock time (left) and peak memory consumption (right) of selected dimensionality reduction methods in dataset R1. Methods that failed to finish are not shown. IDF was used in preprocessing. Exact Hamming distance (for SimHash) or cosine distance (for other methods) was used in k-NN search. Abbreviations: Spectral, spectral embedding; GRP, Gaussian random projection, SRP: sparse random projection; DR, dimensionality reduction.

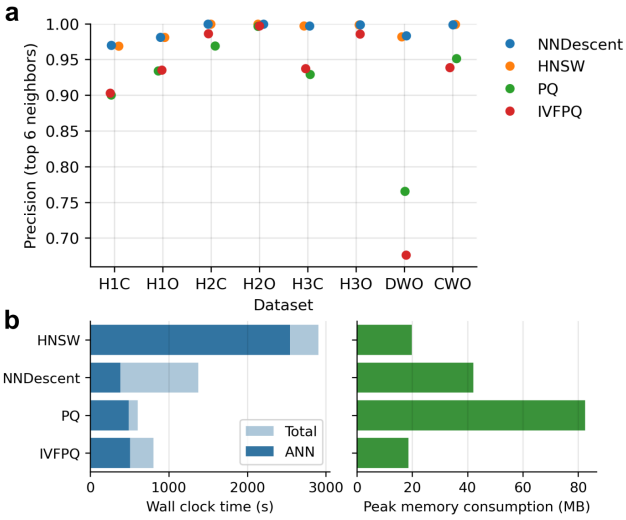

Figure 5. Evaluation of ANN methods.
(a) Dot plot showing the overlap graph precision of various approximate k-NN methods at k = 6. Each color represent a distinct ANN method. **(b)** Wall clock time (left) and peak memory consumption (right) of selected approximate k-NN methods in dataset H7. IDF was used in preprocessing. All methods used cosine similarity for distance computation. Feature matrices were dimensionality reduced to 3,000 dimensions via Sparse Random Projection prior to neighbor search. Abbreviations: HNSW, Hierarchical Navigable Small World; PQ, Product Quantization; IVFPQ, Inverted File with Product Quantization; RPF, Random Projection Forest.

ther confirmed that SRP was the fastest method and had the lowest memory usage among those tested (Figure 4b).

The accuracy of k-NN search after DR typically depends on the embedding dimension, as higher dimensions allow more information from the original feature matrix to be retained. To characterize this relationship, we varied the embedding dimension for each dataset using SRP. The results showed that increasing the embedding dimension improved accuracy, but with diminishing returns (Figures S6, S7, and S8). For smaller datasets (H1 and H2), performance plateaued at around 500 embedding dimensions, while larger datasets (H3) required 3000 or more dimensions to achieve near-maximum precision and overlap size.

To better understand this trend, we examined the correlation between pairwise cosine distances before and after SRP in dataset H7. The results showed a strong positive correlation that increased with embedding dimension, with the Pearson R^2 reaching 0.98 at 3000 dimensions (Figure S9). Based on these findings, we selected SRP as the DR method for further benchmarking due to its scalability, efficiency, and strong distance-preserving properties.

NNDescent offers the best trade-off between speed and accuracy among ANN methods

In addition to dimensionality reduction, another strategy to accelerate k-NN search is to use approximate k-NN (ANN) algorithms, which leverage indexing structures or heuristics to reduce the computational cost of neighbor retrieval. We evaluated five ANN approaches that are widely used in single-cell sequencing and information retrieval applications: NNDescent [24], Hierarchical Navigable Small World (HNSW) [27], product quantization (PQ) [28], Inverted File with Product Quantization (IVF-PQ) [28], and random projection forest (RPF) [29]. All ANN methods were applied to low-dimensional embeddings generated by sparse random projection (SRP).

Among these methods, NNDescent, HNSW, and PQ achieved high precision when used to construct overlap graphs, whereas IVF-PQ and RPF performed less favorably in terms of accuracy (Figure 5a). In terms of computational efficiency, NNDescent and IVF-PQ were the fastest, while RPF consumed the least memory (Figure 5b). Across all configurations, the ANN step emerged as the computational bottleneck in the FEDRANN pipeline, often requiring significantly more time than the SRP step (Figure 5b), although both steps had comparable peak memory usage (Figures 4b and 5b). Therefore, we found NNDescent to be the most useful ANN method for our workflow. It offered the best trade-off between speed and accuracy, making it well-suited for scalable overlap graph construction when combined with SRP-based dimensionality reduction.

High-precision, time-efficient overlap graph construction on the human genome

To validate the performance of the IDF–SRP–NNDescent (ISN) combination on genome-scale datasets and benchmark it against established methods, we constructed overlap graphs for the entire human genome using ISN, based on Oxford Nanopore Technologies (ONT) R9 and PacBio HiFi sequencing data. The resulting graphs were compared to those generated by the widely used alignment tool Minimap2 (in all-vs-all mode) using the same datasets.

Both ISN and Minimap2 achieved high precision across both datasets; however, ISN consistently outperformed Minimap2 in terms of edge precision (Figure 6a). In terms of computational efficiency, ISN also showed a 40% reduction in overall wall clock time, although it incurred higher peak memory usage. These results highlight the practical potential of the FEDRANN framework (realized as SIP) for large-scale *de novo* assembly, offering a favorable balance of accuracy and speed.

Discussions

In this study, we proposed a novel method for overlap graph construction that leverages DR and ANN techniques. We conducted extensive evaluations on both simulated and real long-read sequencing datasets. Among the combinations we tested, the IDF-SPR-

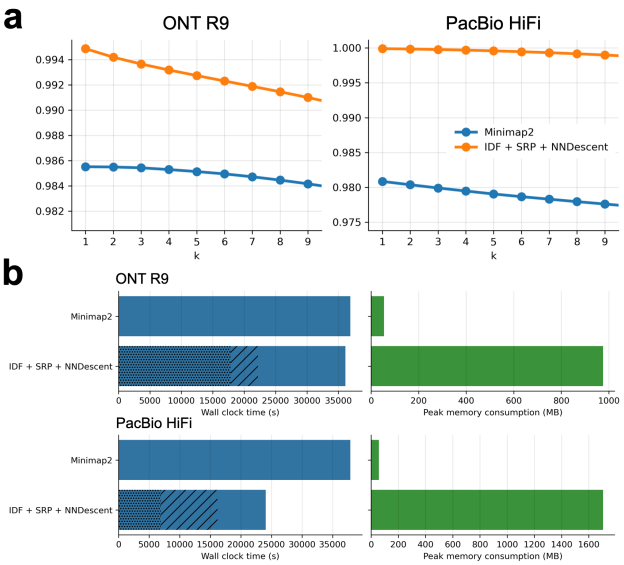

Figure 6. Benchmarking ISN against Minimap2 in the human genome. **(a)** Precision of top k overlaps identified by Minimap2 and ISN method on human genome sequences from ONT R9 (left) and pacbio (right) platform. **(b)** Wall clock time and peak memory consumption for both methods. Abbreviations: ISN, IDF–SRP–NNDescent method.

NNDescent (ISN) pipeline achieved the best trade-off between accuracy and computational efficiency. While SimHash, conceptually similar to SRP, also performed well, it lacked the availability of an efficient implementation. Benchmarking results showed that ISN outperformed Minimap2 in generating more accurate overlap graphs, particularly in complex genomic regions such as the HLA and IGK loci, as well as across the entire human genome. While the concept of treating overlap graph construction as a special case of k-NN search has been introduced in prior work [6], to our knowledge, the explicit application of DR and ANN algorithms to this problem has not been previously reported, and the results above demonstrated the potential of the FEDRANN workflow in finding overlapping sequences.

The ISN implementation used in this work relied primarily on general-purpose libraries not specifically optimized for nucleotide sequence processing. Despite this, ISN was able to construct an overlap graph for the whole human genome in 7 hours—representing a 40% speedup over Minimap2. We anticipate that further improvements, such as re-implementing components in high-performance languages like C++ or Rust, employing multiprocessing for IDF and DR steps, and adopting more efficient data structures, will enhance computational performance and mitigate the memory bottlenecks observed in the current implementation.

A notable limitation of our study is the absence of end-to-end *de novo* assembly results. To maintain focus and clarity, we evaluated overlap graph construction methods based solely on metrics such as precision, mean overlap size, and computational cost. However, the quality of a final *de novo* assembly may depend on additional properties of the overlap graph not captured by our current evaluation framework. For example, consistent graph connectivity across different genomic regions is often critical for achieving high contiguity in the assembled genome, which is difficult to quantify. A key direction for future work is to integrate our approach into a complete OLC-based *de novo* assembly pipeline and to assess the resulting assemblies in terms of contiguity, completeness, and base-level accuracy.

Overlap graph construction can be seen as a special case of the sequence similarity search (SSS) problem. Given a set of query sequences, the goal of the SSS problem is to find sequences that are most similar to each query sequence from a large target database. In principle, our methodology can be readily extended to other sequence similarity search problems after necessary modifications. One application of particular interest is taxonomic classification in metagenomic analyses, where query sequences are typically sequencing reads or assembled contigs, and target sequences come from a pre-indexed database of microbial reference genomes. Unlike overlap graph construction, where all sequences are processed together, taxonomic classification often relies on precomputed indexes to enable efficient querying. Our ISN method is well-suited to such workflows, as both IDF and SRP are linear transformations that can be applied independently to new data, and NNDescent supports incremental ANN search against existing indexes. Nevertheless, further empirical studies are needed to validate the utility of DR and ANN techniques in metagenomic classification and other related sequence analysis tasks.

Methods

Feature extraction

We encoded each input sequencing read as a high-dimensional sparse vector based on its constituent k-mers. The feature extraction process proceeded as follows. First, sequencing reads were loaded from the input FASTQ file, and the reverse complement of each read was computed and included in the dataset. Second, a fixed k-mer alphabet was defined by hashing all k-mers from the reads (both original and reverse complement) using the xxHash function [30] and selecting those with hash values below a predefined threshold. This procedure effectively sampled a subset of the full k-mer space while ensuring uniform selection. Only k-mers present in the selected alphabet were retained for encoding.

For each read, the presence of a retained k-mer was recorded in binary format—set to 1 if the k-mer occurred at least once, and 0 otherwise—resulting in a sparse sequence-by-k-mer feature matrix. This sparse binary matrix was implemented using the Compressed Sparse Row (CSR) format to minimize memory footprint and accelerate subsequent computations. Optionally, the matrix could be transformed using term frequency (TF) and/or inverse document frequency (IDF) weighting to modulate the contribution of each k-mer based on its frequency across the dataset.

The resulting feature matrix was used as input for the dimensionality reduction and approximate nearest neighbor (ANN) search components of the FEDRANN workflow. Unless otherwise specified, we used a k-mer size of 19 and selected approximately 10% of all possible k-mers by setting the hash threshold to 10% of the maximum possible hash value.

Dimensionality reduction

We employed a variety of algorithms for dimensionality reduction, including scBiMapping, UMAP, PCA, GRP, SRP, Spectral Embedding, and SimHash. See Table S2 for implementation details.

k-nearest neighbor search

We employed a variety of exact and approximate algorithms for k-NN search, including brute-force search, HNSW, PQ, IVF-PQ, RPF, and NNDescent. See Table S3 for implementation details.

Overlap graph construction

For a dataset of n sequencing reads, an undirected overlap graph $G = (V, E)$ was constructed from pairs of reads identified as overlapping. Each vertex in the graph represented an oriented read—either in its forward or reverse-complement orientation—resulting in a total of $2n$ vertices. An edge $\{u, v\}$ was added to

the graph if read u was among the k nearest neighbors of read v, or vice versa, indicating that the two reads likely originated from overlapping genomic regions. The total number of edges in the graph ranged from kn to $2kn$, depending on the degree of mutuality among nearest-neighbor relationships (i.e., whether overlaps were reciprocal or one-sided).

Benchmarking datasets

To evaluate various overlap graph construction methods, we used sequencing data from four model species: *Homo sapiens*, *Oryza sativa*, *Caenorhabditis elegans*, and *Drosophila melanogaster*. The corresponding reference genomes were obtained from the following sources: the complete human T2T-CHM13 assembly v2.0 [3], the *Oryza sativa* Nipponbare IRGSP-1.0 assembly [31], the *Caenorhabditis elegans* WBcel235 assembly (WormBase release WS285; NCBI Assembly ID: GCA_000002985.3),and the *Drosophila melanogaster* Release 6 genome assembly (NCBI Assembly ID: GCF_000001215.4).

Simulated sequencing datasets with varying read lengths, error rates, and coverage depths were generated using PBSIM3 [32], based on the `ERRHMM-ONT-HQ` model. Real sequencing datasets from Oxford Nanopore Technologies (ONT), CycloneSEQ, and PacBio HiFi platforms were obtained from publicly available databases. ONT R10 sequencing data for HG002 was downloaded from https://epi2me.nanoporetech.com/gm24385_2020.11/. ONT R9 sequencing data for *C. elegans* (SRR10028111) and *D. melanogaster* (SRR13070625) were retrieved from the NCBI Sequence Read Archive (SRA). CycloneSEQ G400-ER sequencing data for HG002 was generated in-house using genomic DNA extracted from the HG002 lymphoid cell line following standard library preparation and sequencing protocols. PacBio HiFi sequencing data for the HG002 sample was downloaded from the Genome in a Bottle (GIAB) project [33] (https://ftptrace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data_indexes/AshkenazimTrio/sequence.index.AJtrio_PacBio_CCS_15kb_10022018.HG002).

Six genomic regions of varying size and complexity were selected for benchmarking (Table S1). These included: (1) the human *HLA* immunogene cluster and the *IGK* immunoglobulin κ-light chain locus, both of which contain more than 40% repetitive elements and were of medical research interest; (2) human chromosome 22, which includes extended heterochromatic segments. For each region, real sequencing data were mapped to the corresponding reference genome using Minimap2 with the following parameters: `-k 19 -w 5 -A 3 -B 2 -m 250 --secondary=no`, and reads belonging to the given region were extracted.

Overlap graph evaluation

For each benchmarking dataset, overlap graphs constructed using nearest neighbors identified by a specific method were evaluated against a reference graph $G' = (V, E')$ built from the same dataset. The reference graph shared the same set of vertices V as the overlap graphs. For real sequencing data, the reference edges E' were defined based on the alignment positions of reads in the reference genome. Sequencing reads were filtered based on the following criteria to remove ambiguously aligned reads: (1) read length ≥ 5 kb; (2) aligned fraction $\geq 50\%$; (3) mapping quality ≥ 30. For simulated data, the edges E' were determined according to the genomic intervals from which each read was simulated, as reported by PBSIM3.

For each edge $\{u, v\}$ identified in overlap graph $G = (V, E)$, if the same edge exist in the reference graph G', this edge was considered correct and the overlap size between reads u and v were recorded. Otherwise, this edge was considered incorrect and its overlap size treated as zero. We used four metrics to quantitatively evaluate the quality of an overlap graph: precision, mean overlap size, singleton percentage and connected components number. Precision was defined as the number of correct edges divided by the number of total edges. Mean overlap size was defined as the arithmetic mean of overlap size of all edges. The overlap size of incorrect edges were considered zero for this calculation. Singleton percentage was defined as the proportion of isolated nodes (singletons) relative to all nodes in the overlap graph after false edges are removed. This metric essentially quantifies the percentage of reads that do not have any correct nearest neighbors. Connected components number was defined as the number of connected components in the overlap graph after the false edges are removed. Both metrics - singleton percentage and connected components count - serve as indicators of the overall connectivity quality in the generated overlap graph.

For benchmarking purposes, we also generated overlap graphs using six state-of-the-art tools: Minimap2 (version 2.24), xRead (version 1.0.0), BLEND (v1.0.0), MHAP (version 2.1.1), MECAT2 (v20190314), and wtdbg2 (version 2.5). All alignments were filtered using two criteria: (1) ≥ 100 matched bases and (2) $\geq 10\%$ alignment identity. Notably, for Minimap2 PAF files (which produced exceptionally large outputs exceeding 4TB for human whole-genome sequencing read alignments), we increased the identity threshold to 30% to reduce computational overhead. For each read, top k longest overlaps were selected for overlap graph construction. Refer to supplementary table for detail parameters of these tools.

Data availability

CycloneSEQ G400-ER sequencing data generated in this study will be deposited to the publicly available database CNGBdb.

Ethics statement

This study was conducted in adherence to ethical standards to ensure the responsible and respectful treatment of all data and materials. Ethical approval for this study was obtained from the Institutional Review Board of BGI (FT 20060, FT 17099).

Acknowledgments

This work was supported by National Key R&D Program of China (No.2024YFC3406300) and "Pioneer" and "Leading Goose" R&D Program of Zhejiang (No.2024C03004).

Competing interests

The authors have submitted patent applications related to the methods or results presented in this manuscript.

References

1. Sohn Ji, Nam JW. The present and future of de novo whole-genome assembly. Briefings in bioinformatics 2018;19(1):23–40.
2. Zhang JY, Zhang Y, Wang L, Guo F, Yun Q, Zeng T, et al. A single-molecule nanopore sequencing platform. bioRxiv 2024;p. 2024–08.
3. Nurk S, Koren S, Rhie A, Rautiainen M, Bzikadze AV, Mikheenko A, et al. The complete sequence of a human genome. Science 2022;376(6588):44–53.
4. Espinosa E, Bautista R, Larrosa R, Plata O. Advancements in long-read genome sequencing technologies and algorithms. Genomics 2024;p. 110842.

5. Koren S, Walenz BP, Berlin K, Miller JR, Bergman NH, Phillippy AM. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. Genome research 2017;27(5):722–736.
6. Shafin K, Pesout T, Lorig-Roach R, Haukness M, Olsen HE, Bosworth C, et al. Nanopore sequencing and the Shasta toolkit enable efficient de novo assembly of eleven human genomes. Nature biotechnology 2020;38(9):1044–1053.
7. Nie F, Ni P, Huang N, Zhang J, Wang Z, Xiao C, et al. De novo diploid genome assembly using long noisy reads. Nature Communications 2024;15(1):2964.
8. Rizzi R, Beretta S, Patterson M, Pirola Y, Previtali M, Della Vedova G, et al. Overlap graphs and de Bruijn graphs: data structures for de novo genome assembly in the big data era. Quantitative Biology 2019;7:278–292.
9. Altschul S. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Bioinformatics 2012;298:3389.
10. Myers G. Efficient local alignment discovery amongst noisy long reads. In: International Workshop on Algorithms in Bioinformatics Springer; 2014. p. 52–67.
11. Li H. Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics 2018;34(18):3094–3100.
12. Vinga S, Almeida J. Alignment-free sequence comparison—a review. Bioinformatics 2003;19(4):513–523.
13. Kong T, Wang Y, Liu B. xRead: a coverage-guided approach for scalable construction of read overlapping graph. GigaScience 2025;14:giaf007.
14. Chen Y, Nie F, Xie SQ, Zheng YF, Dai Q, Bray T, et al. Efficient assembly of nanopore reads via highly accurate and intact error correction. Nature Communications 2021;12(1):60.
15. Berlin K, Koren S, Chin CS, Drake JP, Landolin JM, Phillippy AM. Assembling large genomes with single-molecule sequencing and locality-sensitive hashing. Nature biotechnology 2015;33(6):623–630.
16. Broder AZ. On the resemblance and containment of documents. In: Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171) IEEE; 1997. p. 21–29.
17. Greenberg GC. Analysis and applications of k-mer based methods in bioinformatics. PhD thesis, University of Illinois at Urbana-Champaign; 2023.
18. Li H. Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences. Bioinformatics 2016;32(14):2103–2110.
19. Charikar MS. Similarity estimation techniques from rounding algorithms. In: Proceedings of the thiry-fourth annual ACM symposium on Theory of computing; 2002. p. 380–388.
20. Ponsero AJ, Miller M, Hurwitz BL. Comparison of k-mer-based de novo comparative metagenomic tools and approaches. Microbiome research reports 2023;2(4):27.
21. Miller JR, Delcher AL, Koren S, Venter E, Walenz BP, Brownley A, et al. Aggressive assembly of pyrosequencing reads with mates. Bioinformatics 2008;24(24):2818–2824.
22. Dunteman GH. Principal components analysis, vol. 69. Sage; 1989.
23. McInnes L, Healy J, Melville J. Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:180203426 2018;.
24. Dong W, Moses C, Li K. Efficient k-nearest neighbor graph construction for generic similarity measures. In: Proceedings of the 20th international conference on World wide web; 2011. p. 577–586.
25. Firtina C, Park J, Alser M, Kim JS, Cali DS, Shahroodi T, et al. BLEND: a fast, memory-efficient and accurate mechanism to find fuzzy seed matches in genome analysis. NAR Genomics and Bioinformatics 2023;5(1):lqad004.
26. Li J, Tian Y, Wang Y, Jin L. dna2bit: high performance genomic distance estimation software for microbial genome analysis. Frontiers in Microbiology 2024;15:1521181.
27. Malkov YA, Yashunin DA. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. IEEE transactions on pattern analysis and machine intelligence 2018;42(4):824–836.
28. Jegou H, Douze M, Schmid C. Product quantization for nearest neighbor search. IEEE transactions on pattern analysis and machine intelligence 2010;33(1):117–128.
29. Yan D, Wang Y, Wang J, Wang H, Li Z. K-nearest neighbor search by random projection forests. IEEE Transactions on Big Data 2019;7(1):147–157.
30. xxHash;. https://xxhash.com/.
31. Kawahara Y, de la Bastide M, Hamilton JP, Kanamori H, McCombie WR, Ouyang S, et al. Improvement of the Oryza sativa Nipponbare reference genome using next generation sequence and optical map data. Rice 2013;6:1–10.
32. Ono Y, Hamada M, Asai K. PBSIM3: a simulator for all types of PacBio and ONT long reads. NAR genomics and bioinformatics 2022;4(4):lqac092.
33. Zook JM, Hansen NF, Olson ND, Chapman L, Mullikin JC, Xiao C, et al. A robust benchmark for detection of germline large deletions and insertions. Nature biotechnology 2020;38(11):1347–1355.

Supplementary information

Supplementary figures

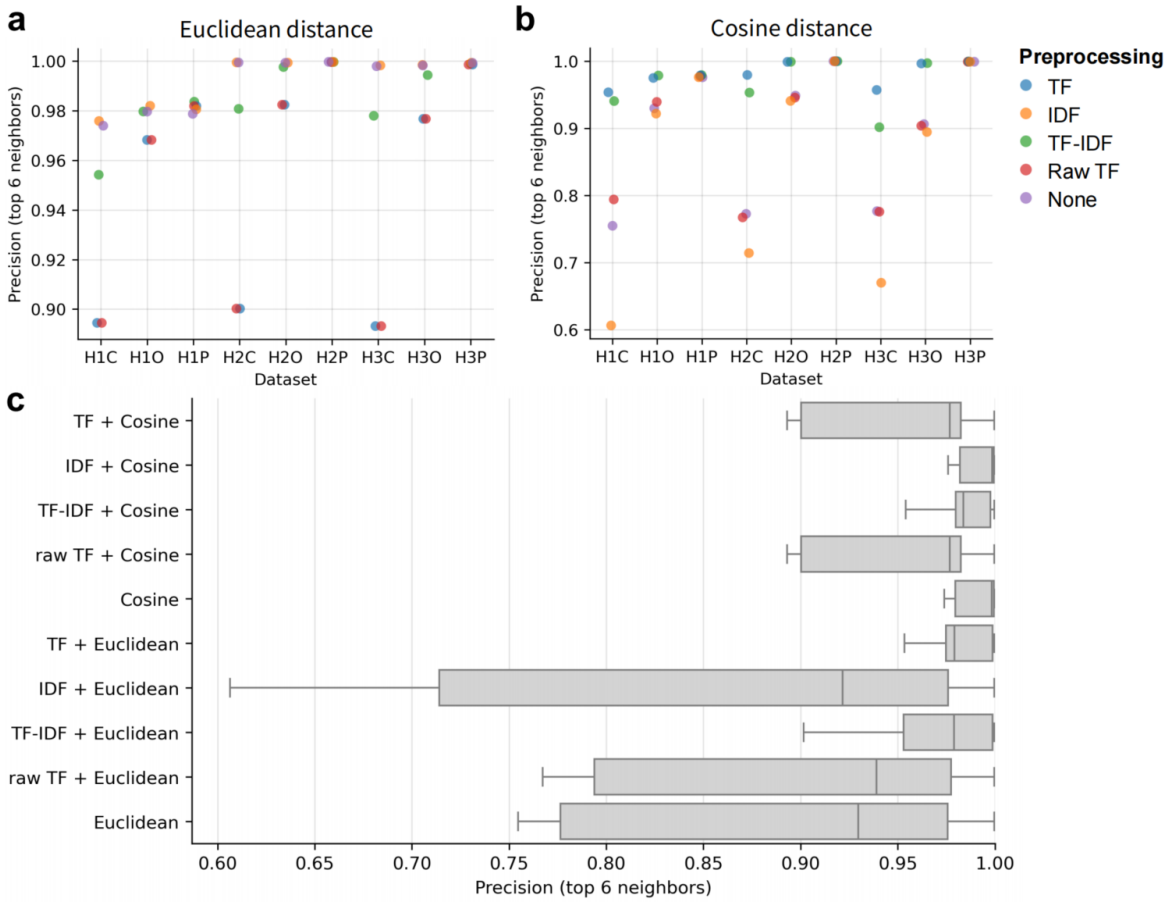

Figure S1. Evaluation of preprocessing methods and distance metrics (summary).
(a-b) Precision of top six neighbors of various preprocessing methods combined with Euclidean **(a)** or cosine **(b)** distance for various datasets. **(c)** Same as **(a-b)**, with precision grouped by method for clarity. In **(a)-(c)**, no dimensionality reduction or ANN methods were used.

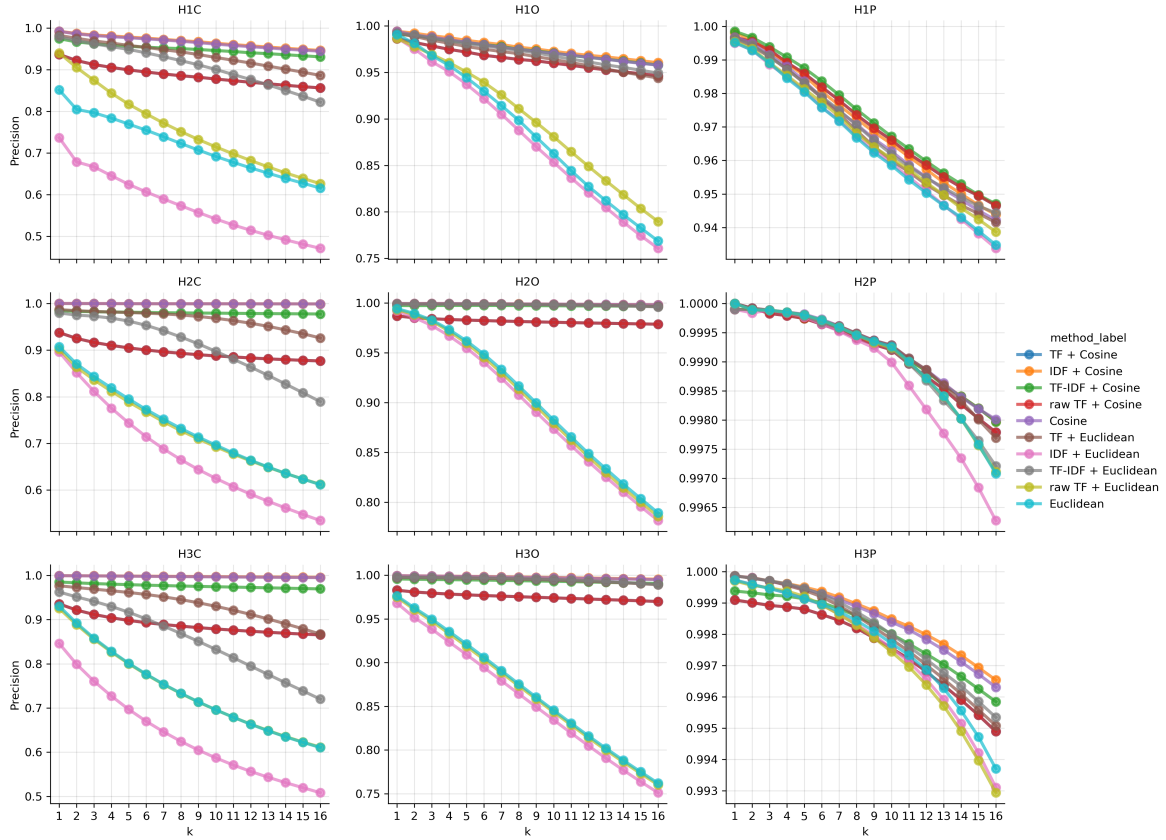

Figure S2. Precision of preprocessing methods and distance metrics.
Precision of various preprocessing methods combined with Euclidean or cosine distance for various datasets. No dimensionality reduction or ANN methods were used.

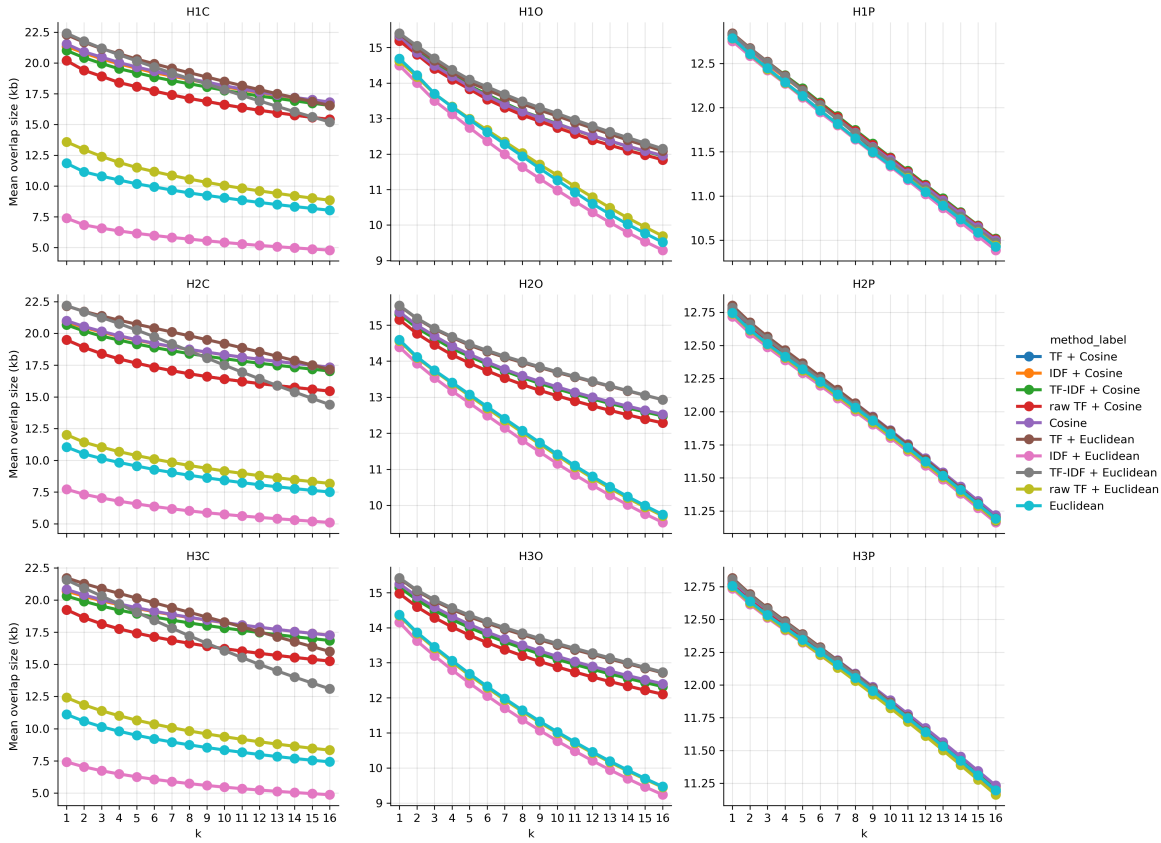

Figure S3. Mean overlap sizes of preprocessing methods and distance metrics.
Mean overlap sizes of various preprocessing methods combined with Euclidean or cosine distance for various datasets. No dimensionality reduction or ANN methods were used.

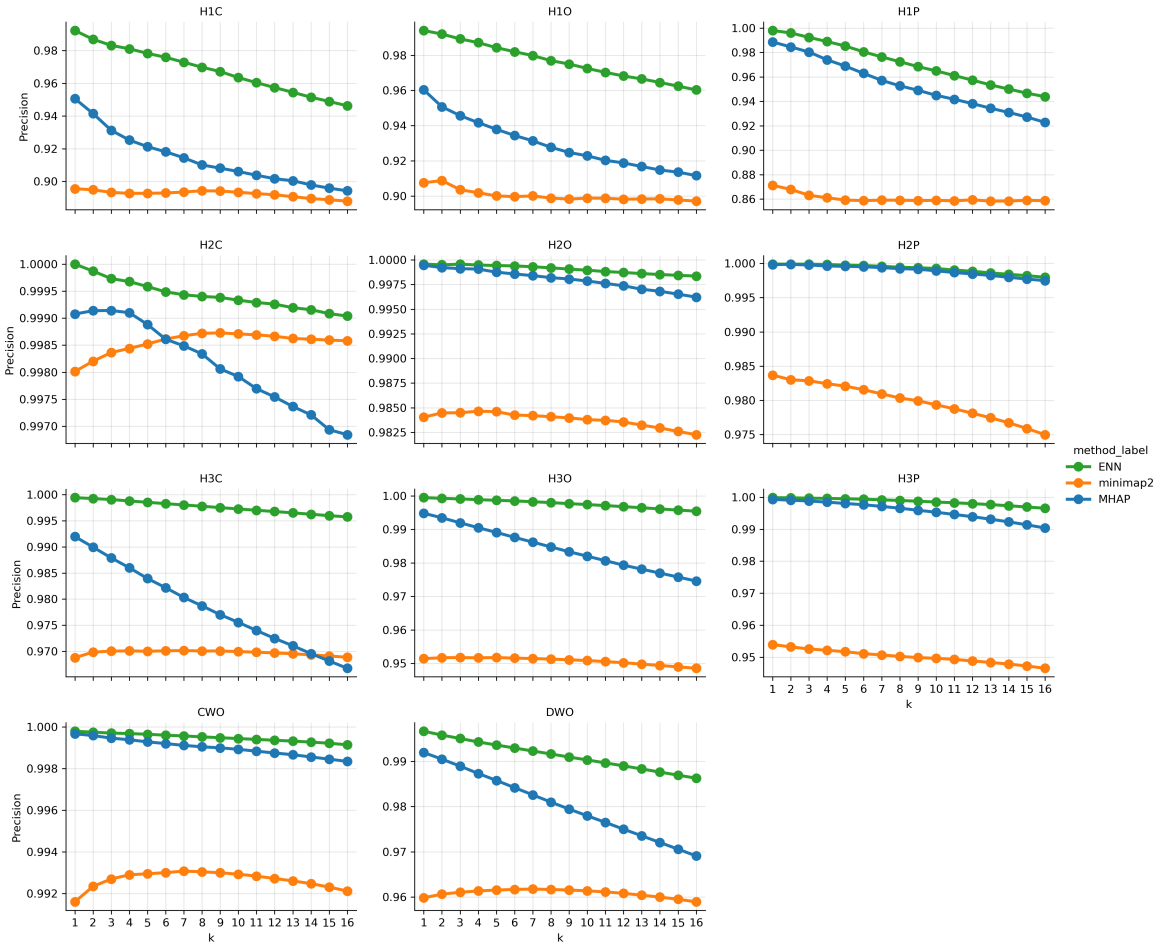

Figure S4. Precision of overlap graph constructed by seed-and-extension, MinHash and exact k-nearest neighbors.
Precision of overlap graph constructed by seed-and-extension, MinHash and exact k-nearest neighbors for various datasets.

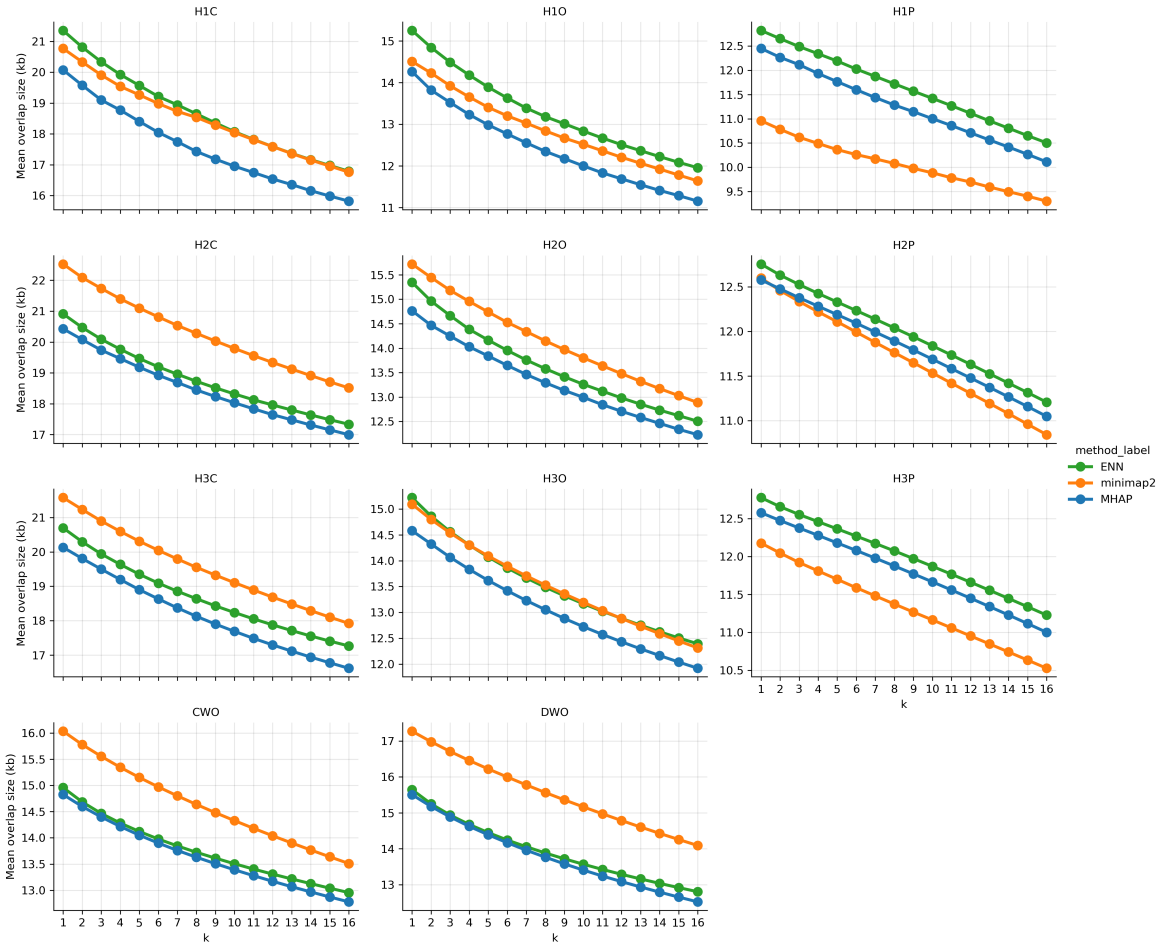

Figure S5. Mean overlap sizes of overlap graph constructed by seed-and-extension, MinHash and exact k-nearest neighbors.
Mean overlap sizes of overlap graph constructed by seed-and-extension, MinHash and exact k-nearest neighbors (ENN) for various datasets.

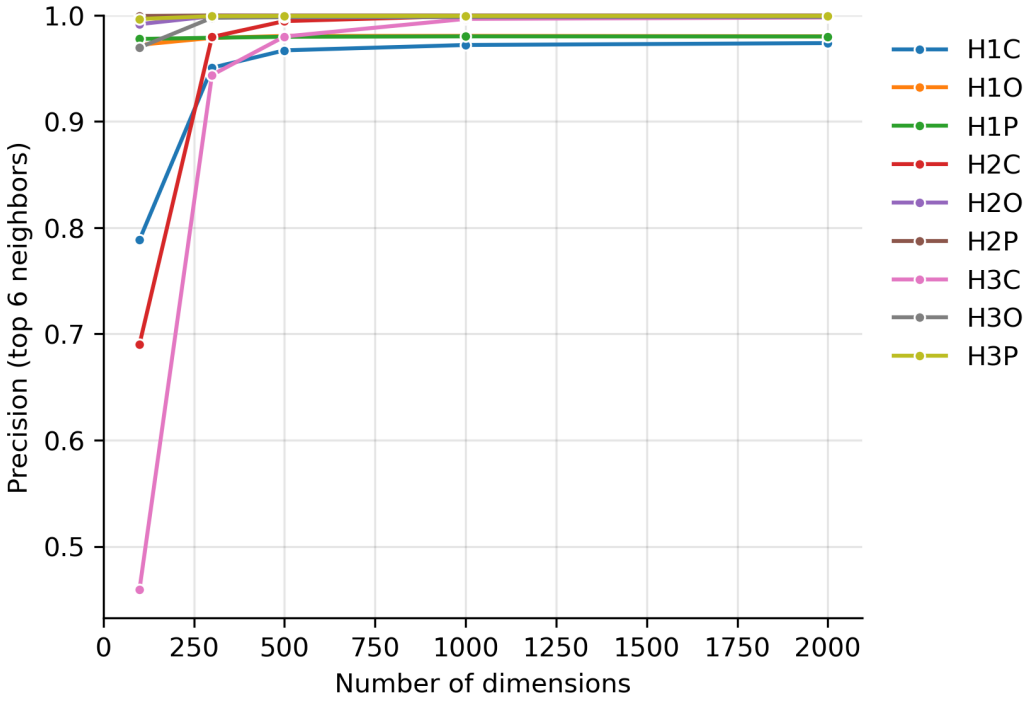

Figure S6. Overlap graph precision across different reduced dimensions (summary).
Precision of k-nearest neighbor overlaps (k=6) across different feature dimensions (100 to 3,000 dimensions) for various datasets. Sparse Random Projection was used as dimensionality recution method. No ANN method was used.

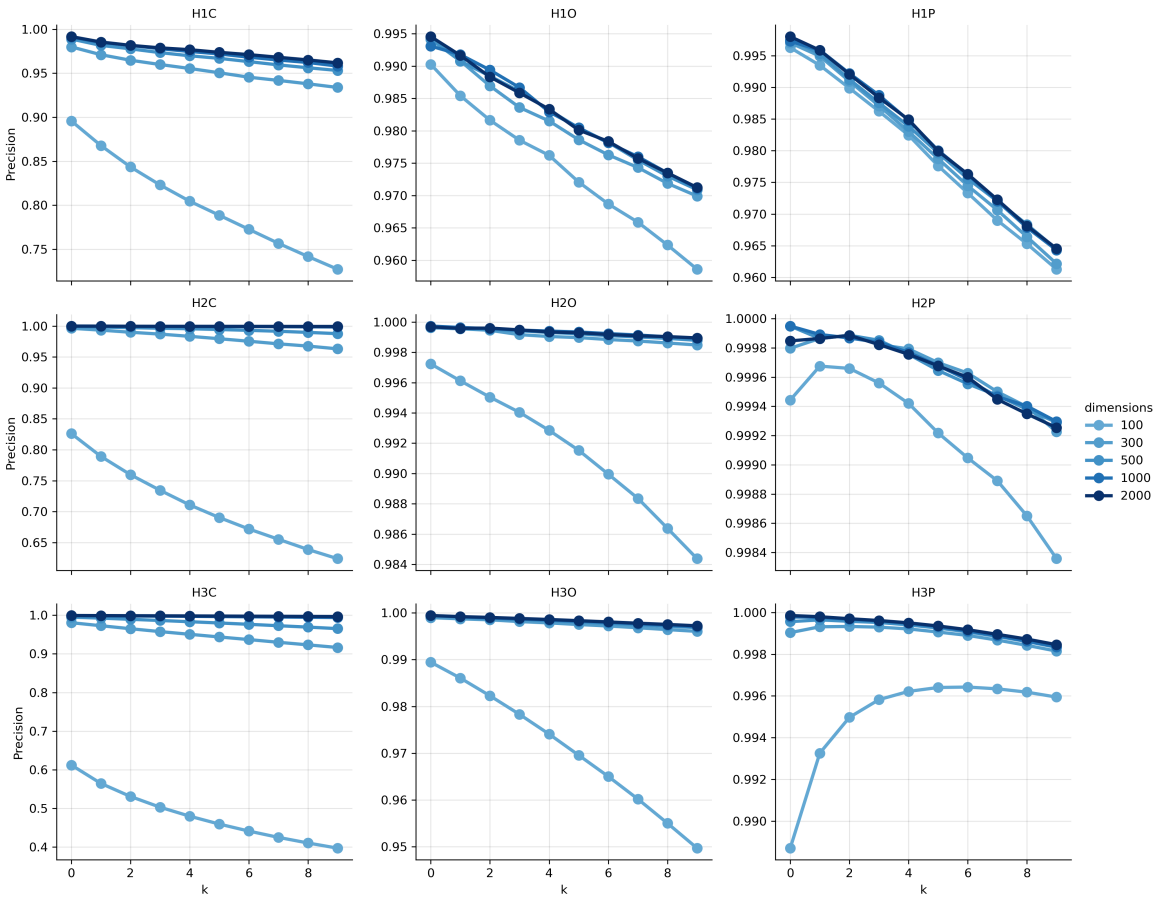

Figure S7. Overlap graph precision across different reduced dimensions.
Precision of k-nearest neighbor overlaps (k=6) across different feature dimensions (100 to 2,000 dimensions) for various datasets. Sparse Random Projection was used as dimensionality recution method. No ANN method was used.

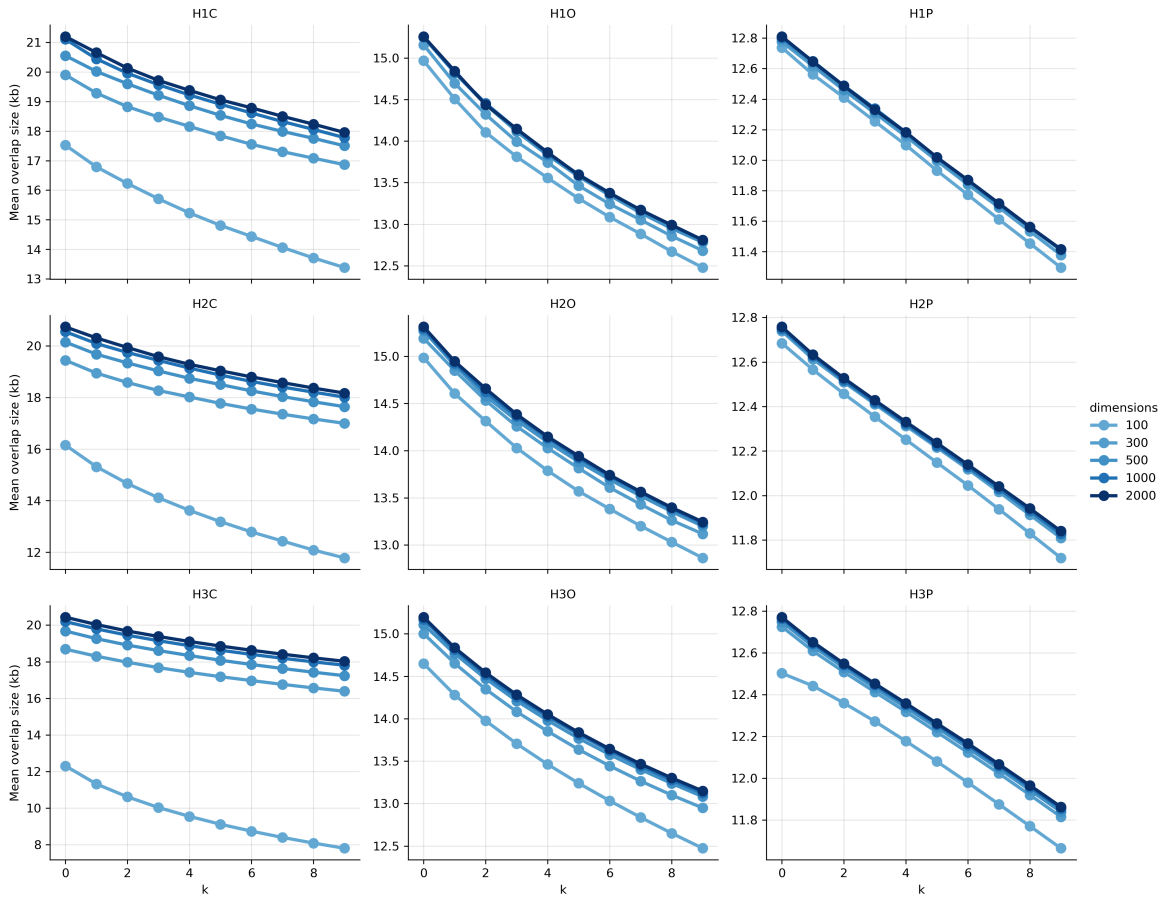

Figure S8. Overlap graph overlap size across different reduced dimensions.
Overlap size of k-nearest neighbor overlaps (k=6) across different feature dimensions (100 to 2,000 dimensions) for various datasets. Sparse Random Projection was used as dimensionality recution method. No ANN method was used.

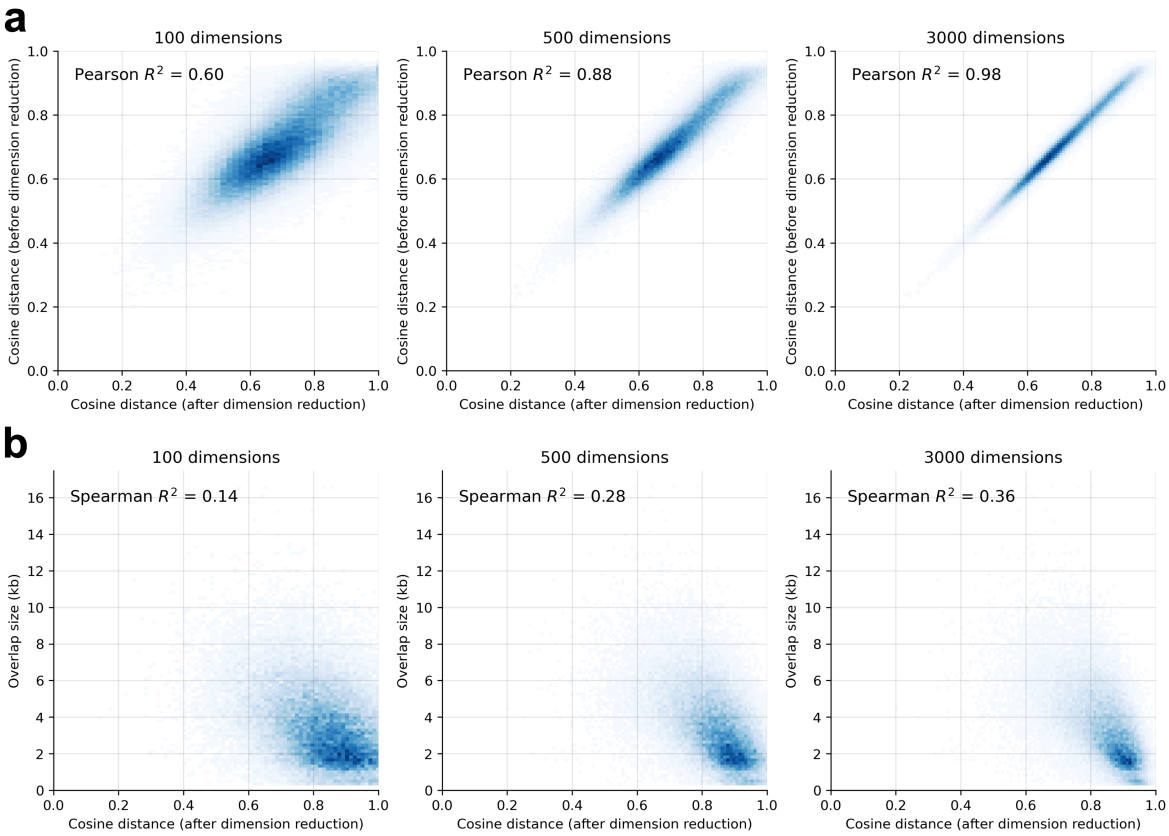

Figure S9. Correlation analysis of distance metrics before/after dimensionality reduction and of cosine distance (after dimensionality reduction) and overlap size. **(a)** Spearman correlation of distance metrics before/after dimensionality reduction at three reduced dimensions (100,500,3000) at dataset human chromosome 1. **(b)** Spearman correlation of cosine distance (after dimensionality reduction) and overlap size at three reduced dimensions (100,500,3000) at human chromosome 1.

Supplementary tables

Table S1. Detail information (region size, reads information, and feature number) of real datasets used in this study.

Sample	Region	Platform	ID	Reference size (Mb)	Read count	Read N50 (kb)
H. sapiens (HG002)	IGK	CycloneSEQ G400-ER	H1C	3.92	2,585	31.99
H. sapiens (HG002)	IGK	ONT R10	H1O	3.92	2,701	22.97
H. sapiens (HG002)	IGK	PacBio HiFi	H1P	3.92	2,624	13.48
H. sapiens (HG002)	HLA	CycloneSEQ G400-ER	H2C	5.75	9,715	31.12
H. sapiens (HG002)	HLA	ONT R10	H2O	5.75	12,576	23.16
H. sapiens (HG002)	HLA	PacBio HiFi	H2P	5.75	12,872	13.42
H. sapiens (HG002)	Chromosome 22	CycloneSEQ G400-ER	H3C	51.32	71,848	30.70
H. sapiens (HG002)	Chromosome 22	ONT R10	H3O	51.32	81,891	22.78
H. sapiens (HG002)	Chromosome 22	PacBio HiFi	H3P	51.32	85,993	13.43
H. sapiens (HG002)	whole genome	CycloneSEQ G400-ER	HWC	3,117.29	5,023,634	31.12
H. sapiens (HG002)	whole genome	ONT R10	HWO	3,117.29	6,226,490	23.06
H. sapiens (HG002)	whole genome	PacBio HiFi	HWP	3,117.29	6,061,187	13.48
C. elegans (strain N2)	whole genome	ONT R9	CWO	100.29	221,513	19.02
D. melanogaster (BDGP genome strain)	whole genome	ONT R9	DWO	143.73	281,881	23.92

Table S2. Summary of dimensionality reduction methods used in this study. Custom Python implementation of SimHash was used as no existing libraries suitable for analyzing biological sequences were found.

Method	Implementation	Version
scBiMapping	`scBiMapping`	v0.1.0
UMAP	`umap-learn`	v0.5.3
PCA	`sklearn.decomposition.PCA`	v1.2.2
Gaussian Random Projection	`sklearn.random_projection.GaussianRandomProjection`	v1.2.2
Sparse Random Projection	`sklearn.random_projection.SparseRandomProjection`	v1.2.2
Spectral Embedding	`snapatac2.tl.spectral`	v2.1.0
SimHash	Custom Python implementation	–

Table S3. Summary of k-nearest neighbor search methods used in this study.

Method	Implementation	Version
brute-force search	`sklearn.neighbors.NearestNeighbors`	v1.2.2
HNSW	`hnswlib`	v0.7.0
PQ	`faiss`	v1.8.0
IVF-PQ	`faiss`	v1.8.0
RPF	`rpforest`	v1.6
NNDescent	`pynndescent`	v0.5.12