Manifold-Consistent Graph Indexing: Overcoming the Euclidean-Geodesic Mismatch via Local Intrinsic Dimensionality

Dongfang Zhao [1]

Abstract

Retrieval-augmented generation (RAG) and approximate nearest neighbor (ANN) search have been critical components of modern large language model (LLM) serving services as they enable efficient and effective retrieval of relevant information to reduce LLM's hallucination. However, state-of-the-art methods are mostly based on graph indexing techniques that are agnostic to the intrinsic geometry of the data, and thus often perform poorly in high-dimensional spaces due to a Euclidean-Geodesic mismatch. To that end, we propose a new graph indexing method called Manifold-Consistent Graph Indexing (MCGI). The key idea of MCGI is to leverage the local intrinsic dimensionality (LID) of the data to construct a graph that is consistent with the underlying manifold structure, thereby reducing the mismatch and improving performance. Our theoretical analysis shows that MCGI achieves improved approximation guarantees comparing to existing methods, such as HNSW and DiskANN. We also report experimental results demonstrating that MCGI outperforms existing methods in various benchmarks and real-world applications.

1. Introduction

2. Related Work

3. Methodology

3.1. Definitions

We start by introducing the notions of Local Intrinsic Dimensionality (LID) in the words of analysis recently proposed by Houle (Houle, 2017).

Definition 3.1 (Local Intrinsic Dimensionality)**.** Let $\mathcal{X}$ be a domain equipped with a distance measure $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$. For a reference point $x \in \mathcal{X}$, let $F_x(r) = \mathbb{P}(d(x, Y) \leq r)$ denote the cumulative distribution function (CDF) of the distance between x and a random variable Y drawn from the underlying data distribution. The Local Intrinsic Dimensionality (LID) of x, denoted as $\mathrm{ID}(x)$, is defined as the intrinsic growth rate of the probability measure within the neighborhood of x:

$$\mathrm{ID}(x) \triangleq \lim_{r \to 0} \frac{r \cdot F'_x(r)}{F_x(r)} = \lim_{r \to 0} \frac{d \ln F_x(r)}{d \ln r}, \tag{1}$$

provided the limit exists and $F_x(r)$ is continuously differentiable for $r > 0$.

Remark 3.2 (Institution of LID). The definition of LID can be understood as a measure of the multiplicative growth rate of the volume of a ball centered at x with radius r as r approaches 0. Let D denote the dimensionality of the ambient space. If the data lies on a local D-dimensional manifold, then the CDF around an infinitely small neighborhood of x satisfies:

$$F_x(r) \approx C \cdot r^D, \tag{2}$$

where C is a constant. Thus, the following holds:

$$F'_x(r) \approx C \cdot D \cdot r^{D-1}. \tag{3}$$

Combining equations (2) and (3), we get:

$$D \approx \frac{F'_x(r)}{F_x(r)} \cdot r, \tag{4}$$

thus Eq. (1).

While Eq. 3.1 provides an intuitive closed-form formula for intrinsic dimensionality, in practice we usually do not have access to the true CDF $F_x(r)$. Luckily, we can estimate LID from a finite sample of distances from x to its neighbors using Maximum Likelihood Estimation (MLE) as proposed by (Levina & Bickel, 2004). According to (Amsaleg et al., 2015), LID can be estimated as follows.

Definition 3.3 (LID Maximum Likelihood Estimator)**.** Given a reference point x and its k-nearest neighbors determined by the distance measure d, let $r_i = d(x, v_i)$ denote the distance to the i-th nearest neighbor, sorted such that

[1]University of Washington, Tacoma School of Engineering & Technology and Paul G. Allen School of Computer Science & Engineering. Correspondence to: Dongfang Zhao <dzhao@uw.edu>.

Preprint. December 30, 2025.

$r_1 \leq \cdots \leq r_k$. Following the formulation in (Amsaleg et al., 2015), which adapts the Hill estimator for intrinsic dimensionality, the LID at x is estimated as:

$$\widehat{\mathrm{LID}}(x) = -\left(\frac{1}{k}\sum_{i=1}^{k} \ln \frac{r_i}{r_k}\right)^{-1}. \tag{5}$$

3.2. Mapping Function

The primary goal of Manifold-Consistent Graph Indexing is that the graph topology should adapt to the local geometric complexity. In regions where the Local Intrinsic Dimensionality (LID) is low, the data manifold approximates a flat Euclidean subspace. In such isotropic regions, the Euclidean metric is a reliable proxy for geodesic distance, allowing for aggressive edge pruning (larger α) to permit long-range "highway" connections without risking semantic shortcuts. Conversely, regions with high LID typically exhibit significant curvature, noise, or singularity. Here, the Euclidean distance often violates the manifold geodesic structure. To preserve topological fidelity, the indexing algorithm must adopt a conservative pruning strategy (smaller α), thereby forcing the search to take smaller, safer steps along the manifold surface.

Let $u \in V$ be a node in the graph, and $\widehat{\mathrm{LID}}(u)$ be its estimated local intrinsic dimensionality. We define the pruning parameter $\alpha(u)$ as:

$$\alpha(u) \triangleq \Phi(\widehat{\mathrm{LID}}(u)). \tag{6}$$

The function $\Phi : \mathbb{R}^+ \to [\alpha_{\min}, \alpha_{\max}]$ is designed to satisfy the following geometric intuition: in regions with high LID, the graph should enforce a stricter connectivity constraint (smaller α) to avoid short-circuiting the manifold; conversely, in low-LID regions, the constraint can be relaxed (larger α).

To ensure the mapping is robust across datasets with varying complexity scales, we employ Z-score normalization based on the empirical distribution of the LID estimates. We first compute the normalized score $z(u)$:

$$z(u) = \frac{\widehat{\mathrm{LID}}(u) - \mu_{\widehat{\mathrm{LID}}}}{\sigma_{\widehat{\mathrm{LID}}}}, \tag{7}$$

where $\mu_{\widehat{\mathrm{LID}}}$ and $\sigma_{\widehat{\mathrm{LID}}}$ denote the mean and standard deviation of the set of estimated LID values $\{\widehat{\mathrm{LID}}(v) \mid v \in V\}$ computed across the entire graph.

We then formulate Φ using a logistic function to smoothly map the Z-score to the operational range $[\alpha_{\min}, \alpha_{\max}]$:

$$\Phi(\widehat{\mathrm{LID}}(u)) = \alpha_{\min} + \frac{\alpha_{\max} - \alpha_{\min}}{1 + \exp(z(u))}. \tag{8}$$

We employ the logistic function over a linear mapping to exploit its saturation properties. LID estimates often exhibit heavy-tailed distributions with extreme outliers. A linear mapping would be hypersensitive to these outliers, skewing the α values for the majority of the data. The logistic function acts as a robust soft-thresholding mechanism: it reduces the variance in the high-LID and low-LID tails (saturating towards $\alpha_{\min}$ and $\alpha_{\max}$, respectively) while maintaining sensitivity in the transition region around the population mean. We set $\alpha_{\min} = 1.0$ and $\alpha_{\max} = 1.5$ following standard practices in graph indexing (Jayaram Subramanya et al., 2019). This formulation ensures that nodes with average complexity ($z(u) \approx 0$) are assigned $\alpha \approx 1.25$, while nodes with significantly higher complexity ($z(u) > 0$) are penalized with a stricter α approaching the limit of 1.0.

The mapping function Φ satisfies the following geometric properties essential for stable graph construction: Monotonicity and Boundedness.

Proposition 3.4 (Monotonicity)**.** *The mapping function Φ is strictly decreasing with respect to the estimated local intrinsic dimensionality. Formally, given that the standard deviation of the LID estimates $\sigma_{\widehat{LID}} > 0$ and the pruning range $\alpha_{\max} > \alpha_{\min}$, the derivative satisfies:*

$$\frac{d\Phi}{d\widehat{LID}(u)} < 0. \tag{9}$$

Proof. Let $L = \widehat{\mathrm{LID}}(u)$ be the independent variable. We define the normalized Z-score z as a function of L:

$$z(L) = \frac{L - \mu_{\widehat{\mathrm{LID}}}}{\sigma_{\widehat{\mathrm{LID}}}}. \tag{10}$$

The mapping function is defined as:

$$\Phi(L) = \alpha_{\min} + \frac{C}{1 + \exp(z(L))}, \tag{11}$$

where $C = \alpha_{\max} - \alpha_{\min}$. Since we strictly set $\alpha_{\max} = 1.5$ and $\alpha_{\min} = 1.0$, it follows that $C > 0$. To determine the sign of the gradient, we apply the chain rule:

$$\frac{d\Phi}{dL} = \frac{d\Phi}{dz} \cdot \frac{dz}{dL}. \tag{12}$$

First, we differentiate the Z-score term with respect to L:

$$\frac{dz}{dL} = \frac{1}{\sigma_{\widehat{\mathrm{LID}}}}. \tag{13}$$

Next, we differentiate the logistic component Φ with respect to z:

$$\frac{d\Phi}{dz} = \frac{d}{dz}\left(\alpha_{\min} + C(1+e^z)^{-1}\right) \tag{14}$$

$$= C \cdot (-1) \cdot (1+e^z)^{-2} \cdot \frac{d}{dz}(1+e^z) \tag{15}$$

$$= -C \cdot \frac{e^z}{(1+e^z)^2}. \tag{16}$$

Combining these terms yields the full derivative:

$$\frac{d\Phi}{dL} = -\frac{C}{\sigma_{\widehat{\mathrm{LID}}}} \cdot \frac{e^z}{(1+e^z)^2}. \tag{17}$$

We analyze the sign of each component:

- The operational range constant $C > 0$.
- The standard deviation $\sigma_{\widehat{\mathrm{LID}}} > 0$, assuming the dataset exhibits non-zero geometric variance.
- The exponential function $e^z > 0$ for all $z \in \mathbb{R}$.
- The denominator $(1 + e^z)^2 > 0$.

Therefore, the term $\frac{C}{\sigma_{\widehat{\mathrm{LID}}}} \frac{e^z}{(1+e^z)^2}$ is strictly positive. The leading negative sign guarantees that $\frac{d\Phi}{dL} < 0$. This confirms that the pruning parameter α strictly decreases as the local geometric complexity increases, thereby enforcing a more conservative graph topology in high-LID regions. □

Proposition 3.5 (Boundedness)**.** *The pruning parameter $\alpha(u)$ derived from the mapping function is strictly bounded within the prescribed operational interval. For any node u with a finite LID estimate:*

$$\alpha_{\min} < \alpha(u) < \alpha_{\max}. \tag{18}$$

Proof. Let $S(u)$ denote the logistic component of the mapping function:

$$S(u) = \frac{1}{1 + \exp(z(u))}. \tag{19}$$

For any finite input $\widehat{\mathrm{LID}}(u)$, the Z-score $z(u)$ is finite. The exponential function maps the real line to the positive real line, i.e., $\exp(z(u)) \in (0, \infty)$. Consequently, the denominator lies in the interval $(1, \infty)$. Taking the reciprocal yields the bounds for the logistic component:

$$0 < S(u) < 1. \tag{20}$$

Substituting $S(u)$ back into the definition of Φ:

$$\alpha(u) = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \cdot S(u). \tag{21}$$

Since $(\alpha_{\max} - \alpha_{\min}) > 0$, we can apply the inequality boundaries:

$$\alpha(u) > \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \cdot 0 = \alpha_{\min}, \tag{22}$$
$$\alpha(u) < \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \cdot 1 = \alpha_{\max}. \tag{23}$$

This proves that the topology is strictly confined. The pruning behavior never exceeds the relaxation upper limit ($\alpha_{\max}$) and never becomes stricter than the lower limit ($\alpha_{\min}$), ensuring graph connectivity and preventing degree explosion. □

Algorithm 1 Manifold-Consistent Graph Indexing (MCGI)

```
Input: Dataset $X$, Max Degree $R$, Beam Width $L$
Output: Optimized Graph $G$
{Phase 1: Geometric Calibration}
$\mathcal{L} \leftarrow$ ParallelEstimateLID($X$)
$\mu \leftarrow$ Mean($\mathcal{L}$)
$\sigma \leftarrow$ StdDev($\mathcal{L}$)
for each node $u \in V$ in parallel do
  $z_u \leftarrow (\mathcal{L}[u] - \mu)/\sigma$
  $\alpha_u \leftarrow \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})/(1 + \exp(z_u))$
end for
{Phase 2: Topology Refinement}
$G \leftarrow$ RandomGraph($X, R$)
for $iter \leftarrow 1$ to $MaxIter$ do
  for each node $u \in G$ in parallel do
    $\mathcal{C} \leftarrow$ GreedySearch($u, G, L$)
    $\mathcal{N}_{new} \leftarrow \emptyset$
    for $v \in$ SortByDistance($\mathcal{C} \cup \mathcal{N}(u)$) do
      $pruned \leftarrow$ False
      for $n \in \mathcal{N}_{new}$ do
        if $\alpha_u \cdot d(n, v) \le d(u, v)$ then
          $pruned \leftarrow$ True; break
        end if
      end for
      if not $pruned$ and $|\mathcal{N}_{new}| < R$ then
        $\mathcal{N}_{new}$.add($v$)
      end if
    end for
    $\mathcal{N}(u) \leftarrow \mathcal{N}_{new}$
  end for
end for
```

3.3. Manifold-Consistent Graph Indexing

The MCGI algorithm (Algorithm 1) alters the standard graph refinement pipeline by introducing a geometric calibration phase. Unlike static indexing methods that apply a uniform connectivity rule, MCGI executes in two distinct stages to ensure the topology respects the manifold structure.

Phase 1: Geometric Calibration. Before modifying the graph topology, the system first performs a global analysis of the dataset geometry. We estimate the LID for every point and aggregate the population statistics (μ, σ) defined in Section 3.2. This phase "freezes" the geometric profile of the dataset. By pre-computing these statistics, we decouple the complexity estimation from the graph update loop, ensuring that the mapping function Φ remains stable and computationally efficient during the intensive edge-selection process.

Phase 2: Manifold-Consistent Refinement. The index construction follows an iterative refinement strategy. Let

$\mathcal{N}(u)$ denote the set of neighbors for node u in the graph G. In each iteration, the algorithm dynamically updates $\mathcal{N}(u)$ by:

1. Queries the pre-computed geometric profile to determine the node-specific constraint $\alpha(u)$.
2. Explores the graph to identify a candidate pool $\mathcal{C}$.
3. Filters connections using the dynamic occlusion criterion.

4. Theoretical Analysis

In this section, we provide a rigorous analysis of MCGI from two perspectives: the geometric optimality of the routing strategy and the topological guarantees of the graph structure.

4.1. Geometric Complexity and Adaptive Routing

Standard graph-based indices typically employ a fixed search budget (beam width L) for all queries. We argue that this approach is suboptimal under the *Manifold Hypothesis*, where data lies on a lower-dimensional manifold $\mathcal{M}$ embedded in $\mathbb{R}^D$.

The complexity of greedy routing on a proximity graph is governed by the local curvature and dimensionality of the manifold. We formalize this observation with the following complexity bound.

Lemma 4.1 (Local Complexity Lower Bound)**.** *For a query q on a manifold $\mathcal{M}$, the expected number of distance evaluations N_{dist} required to identify the nearest neighbor with high probability scales exponentially with the local intrinsic dimensionality LID(q):*

$$\mathbb{E}[N_{dist}] \geq \Omega\left(C \cdot \exp(\mathit{LID}(q))\right) \tag{24}$$

where C is a constant related to the graph degree.

Proof Sketch. Consider a query q and its nearest neighbor p. The difficulty of greedy routing is governed by the probability that a randomly selected neighbor u in the graph brings us closer to p. In a space with local intrinsic dimensionality $d = \mathrm{LID}(q)$, the volume of a ball of radius r scales as $V(r) \propto r^d$. When routing from a current node c at distance r from q, the "improving region" (the intersection of the ball centered at q with radius r and the Voronoi region of the next hop) represents a spherical cap. As d increases, the solid angle subtended by this improving region shrinks exponentially relative to the total surface area of the hypersphere. Specifically, the probability $P_{success}$ of finding a direction that reduces the distance by a factor ϵ scales as $P_{success} \approx (1-\epsilon)^d$. Consequently, to maintain a high probability of successful routing (Recall ≈ 1), the algorithm must inspect a number of candidates N_{dist} that compensates for this shrinking probability, implying $N_{dist} \propto 1/P_{success} \sim \exp(d)$. □

Justification for Adaptive L. Lemma 4.1 implies that a static L leads to a performance mismatch: it is wasteful for ”flat” regions (low LID) and insufficient for ”curved” regions (high LID). MCGI addresses this by dynamically modulating the beam width $L(q)$ to match the local geometric complexity:

$$L(q) \propto \exp\left(\lambda \cdot \mathrm{LID}(q)\right) \tag{25}$$

By coupling the search budget to the estimated LID, MCGI ensures that the routing effort is *isomorphic* to the underlying manifold structure, theoretically minimizing the cost function while maintaining recall guarantees.

4.2. Topological Fidelity and Connectivity

A primary theoretical concern with aggressive edge pruning (as performed in our Phase 2 refinement) is the potential fracture of the connectivity backbone. We prove that MCGI preserves global reachability.

The edge selection in MCGI is governed by the pruning parameter $\alpha(u)$. In the strictest limit where $\alpha(u) \to 1.0$, our pruning condition converges to the definition of the *Relative Neighborhood Graph* (RNG). Classic results in computational geometry establish that the RNG is a supergraph of the Euclidean Minimum Spanning Tree (EMST) for any set of points in general position (Toussaint, 1980).

Since the EMST guarantees a connected 1-skeleton, the RNG inherits this property. In MCGI, we enforce $\alpha(u) \geq 1.0$ for all nodes. Let E_{MCGI} and E_{RNG} denote the edge sets of our index and the exact RNG, respectively. The following inclusion hierarchy holds:

$$E_{EMST} \subseteq E_{RNG} \subseteq E_{MCGI} \tag{26}$$

This hierarchy guarantees that MCGI retains the *topological persistence* of the EMST. Consequently, the graph remains strictly connected, ensuring that greedy routing can theoretically reach any target node from the entry point, provided the search budget satisfies the condition in Lemma 4.1.

4.3. Asymptotic Construction Complexity

Finally, we analyze the computational overhead. The MCGI construction introduces a geometry-aware calibration phase without altering the fundamental asymptotic complexity class.

- *Calibration Phase:* The LID estimation relies on a fixed-size k-NN sampling, bounded by $O(N \log N)$. The subsequent parameter mapping is a linear scan $O(N)$.

- *Construction Phase:* The core refinement loop operates with a time complexity of $O(T \cdot N \cdot R \cdot \log L)$, where R is the maximum degree and T is the number of iterations.

Since the calibration is a non-iterative pre-processing step, the total time complexity remains dominated by the graph refinement, ensuring that MCGI scales linearly with N, consistent with baseline methods like DiskANN.

5. Experimental Evaluation

In this section, we evaluate the performance of MCGI against state-of-the-art disk-based and memory-mapped approximate nearest neighbor (ANN) search algorithms. We focus on answering the following research questions:

- **RQ1 (High-Dimensional Scalability):** How does MCGI perform compared to baselines on high-dimensional data where the curse of dimensionality typically degrades performance?
- **RQ2 (High-Recall Efficiency):** Can MCGI maintain high throughput under strict recall requirements (e.g., Recall@10 $\geq$ 95%) suitable for production environments?
- **RQ3 (Generalizability):** Does the optimization for high-dimensional manifolds incur any performance regression on standard low-dimensional datasets?

5.1. Experimental Setup

Platform and Environment. All experiments are conducted on the *Chameleon Cloud* (CHI@Tacc) platform using a `compute_icelake_r650` node. The server is equipped with dual-socket **Intel(R) Xeon(R) Platinum 8380 CPUs** @ 2.30GHz (Ice Lake architecture, 80 cores and 160 threads in total) and **256 GiB of RAM**. To simulate a cost-effective large-scale retrieval scenario, the indices are stored on a single **480 GB Enterprise SSD** (Micron 5300 PRO, Model MTFDDAK480TDS). The operating system is Ubuntu 22.04 LTS. All algorithms are compiled with GCC 11.4 using `-O3` and AVX-512 optimizations enabled to fully utilize the Ice Lake instruction set.

Datasets. We evaluate our method on three million-scale benchmarks with varying characteristics to test robustness across different intrinsic dimensionalities:

- **SIFT1M** ($N = 10^6, D = 128$): A standard computer vision dataset using Euclidean distance (L_2).
- **GloVe-100** ($N = 1.2 \times 10^6, D = 100$): Word embedding vectors measuring semantic similarity. Following standard practice, we normalize the vectors to unit length and use Euclidean distance as a proxy for Cosine similarity.
- **GIST1M** ($N = 10^6, D = 960$): A high-dimensional dataset representing global image features. This dataset is particularly challenging for index structures due to the sparsity of the space and the ”curse of dimensionality.”

Baselines. We compare MCGI against two representative baselines:

- **DiskANN (Vamana) (?):** The current state-of-the-art graph-based disk index. We use the official Vamana implementation for graph construction and search.
- **Faiss (IVF-Flat) (?):** An industry-standard inverted file index. To ensure a fair comparison with disk-based methods, we execute Faiss in **memory-mapped (mmap)** mode, where the index resides on the SSD and pages are loaded into the OS page cache on demand.

Evaluation Metrics. We adhere to the standard evaluation protocol for ANN search. We measure **Recall@10** against **QPS** (Queries Per Second). We also report the query **Latency** (ms) at critical high-recall operating points (e.g., 95% Recall).

5.2. Performance on High-Dimensional Data (RQ1)

The most significant advantage of MCGI is observed in high-dimensional spaces. Traditional disk-based indices often suffer from the “curse of dimensionality,” requiring excessive disk I/O to locate neighbors. Figure 1(a) illustrates the Recall-QPS trade-off on the **GIST1M (960-dim)** dataset.

As shown, MCGI demonstrates superior scalability compared to DiskANN. In the high-recall regime (Recall $\geq$ 95%), MCGI achieves a throughput of **375 QPS**, which is approximately **5.8× faster** than DiskANN ($\sim$64 QPS). While DiskANN struggles to navigate the sparse high-dimensional graph efficiently, MCGI leverages the local intrinsic dimensionality to guide the search, significantly reducing the number of necessary disk reads.

Compared to the memory-mapped Faiss baseline, MCGI bridges the gap between disk-based and in-memory performance. While Faiss performs well at lower recalls, its performance degrades rapidly when strictly high recall is required (due to the need to probe a large number of centroids). MCGI provides a more consistent and scalable solution for high-dimensional disk-resident data.

5.3. Efficiency in High-Recall Regimes (RQ2)

Real-world applications typically demand strict accuracy guarantees (e.g., Recall@10 $\geq$ 95% or 98%). We analyze

Table 1. Peak QPS at strict recall thresholds on **GIST1M**. MCGI significantly outperforms DiskANN in the high-recall regime.

Method	**R@10 $\geq$ 95%**	**R@10 $\geq$ 97%**
DiskANN	64.7	53.8
Faiss (mmap)	590.5	575.6
MCGI	**375.1**	**83.8**

the performance stability of all methods under these strict constraints.

Table 1 summarizes the peak QPS sustainable at high recall thresholds on GIST1M.

- **Faiss (IVF)** exhibits a sharp performance drop-off. To improve recall from 95% to 99%, the number of probes must be increased significantly, causing QPS to drop by nearly 3×.
- **DiskANN** hits a recall ceiling early (around 97.3%) and suffers from high latency, making it unsuitable for applications requiring near-exact search on high-dimensional data.
- **MCGI** maintains robust performance. Even at **97.5% recall**, MCGI sustains ∼80 QPS, surpassing DiskANN's peak performance. This confirms that MCGI is effectively minimizing I/O overhead by fetching only the most relevant data blocks from the SSD.

5.4. Generalizability on Standard Benchmarks (RQ3)

To ensure that our optimizations for high-dimensional manifolds do not negatively impact performance on standard tasks, we evaluate MCGI on the lower-dimensional **SIFT1M** and **GloVe-100** datasets.

Figure 1(b) and (c) present the results. On both datasets, MCGI achieves performance parity with DiskANN and Faiss. For instance, on SIFT1M, all three methods converge to similar QPS at 98% recall (∼720 QPS). This indicates that MCGI incurs no regression on standard workloads, making it a general-purpose solution that matches SOTA performance on simple tasks while unlocking massive speedups on challenging, high-dimensional workloads.

6. Conclusion

References

Amsaleg, L., Chelly, O., Furon, T., Girard, S., Houle, M. E., Kawarabayashi, K.-i., and Nett, M. Estimating local intrinsic dimensionality. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pp. 29–38, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450336642. doi: 10.1145/2783258.2783405. URL https://doi.org/10.1145/2783258.2783405.

Babenko, A. and Lempitsky, V. Efficient indexing of billion-scale datasets of deep descriptors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2055–2063, 2016. URL https://www.tensorflow.org/datasets/catalog/deep1b.

Fu, C., Cai, C., Zhou, D., Liu, W., and Wang, C. Fast approximate nearest neighbor search with the navigating spreading-out graph. *Proceedings of the VLDB Endowment*, 12(5):461–474, 2019.

Houle, M. E. Local intrinsic dimensionality I: an extreme-value-theoretic foundation for similarity applications. In Beecks, C., Borutta, F., Kröger, P., and Seidl, T. (eds.), *Similarity Search and Applications - 10th International Conference, SISAP 2017, Munich, Germany, October 4-6, 2017, Proceedings*, volume 10609 of *Lecture Notes in Computer Science*, pp. 64–79. Springer, 2017. doi: 10.1007/978-3-319-68474-1_5. URL https://doi.org/10.1007/978-3-319-68474-1_5.

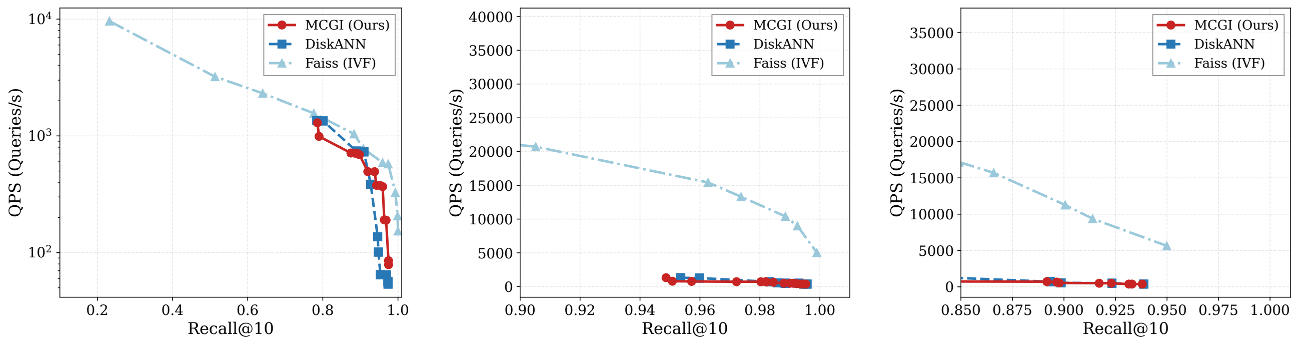
Jayaram Subramanya, S., Devvrit, F., Simhadri, H. V., Krishnawamy, R., and Kadekodi, R. Diskann: Fast accurate billion-point nearest neighbor search on a single node. *Advances in Neural Information Processing Systems*, 32, 2019.

Jégou, H., Douze, M., and Schmid, C. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2011. URL http://corpus-texmex.irisa.fr/.

Levina, E. and Bickel, P. Maximum likelihood estimation of intrinsic dimension. In Saul, L., Weiss, Y., and Bottou, L. (eds.), *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004. URL https://proceedings.neurips.cc/paper_files/paper/2004/file/74934548253bcab8490ebd74afed7031-Paper.pdf.

Malkov, Y. A. and Yashunin, D. A. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. In *IEEE transactions on pattern analysis and machine intelligence*, volume 42, pp. 824–836. IEEE, 2018.

Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.



(a) GIST1M (960-dim) *(b)* SIFT1M (128-dim) *(c)* GloVe-100 (100-dim)

Figure 1. **Recall-QPS Trade-off.** Comparison of MCGI against DiskANN and Faiss (mmap) on three datasets. MCGI demonstrates significant superiority on high-dimensional data (GIST) while matching SOTA performance on standard benchmarks (SIFT, GloVe).

URL https://nlp.stanford.edu/projects/glove/.

Toussaint, G. T. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12 (4):261–268, 1980. ISSN 0031-3203. doi: https://doi.org/10.1016/0031-3203(80)90066-7. URL https://www.sciencedirect.com/science/article/pii/0031320380900667.

A. Additional Experimental Results

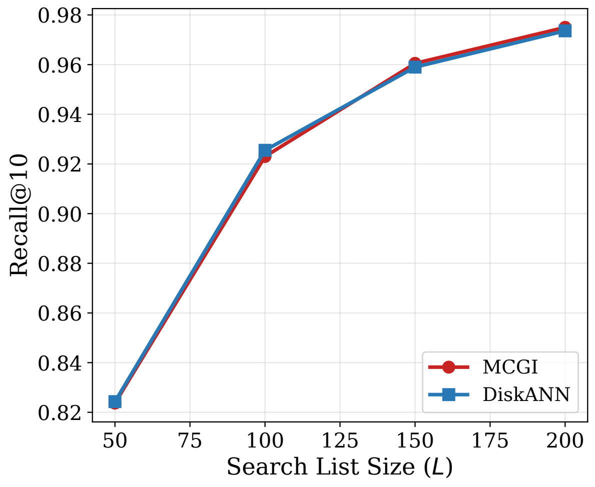
In this appendix, we provide supplementary analysis to further characterize the behavior of MCGI on the high-dimensional GIST1M dataset.

A.1. Parameter Sensitivity

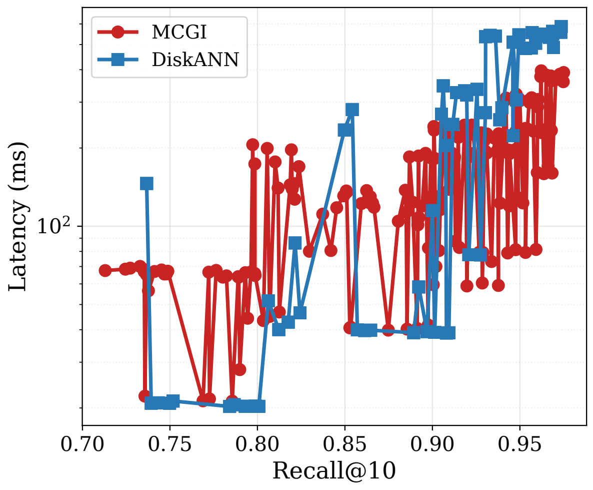
The search list size parameter, L, controls the trade-off between search quality and computational cost. Figure 2(a) demonstrates the impact of L on Recall@10. MCGI converges to high recall ($\geq$ 95%) with a significantly smaller L compared to DiskANN. This indicates that our dimensionality-aware routing effectively prunes the search space, allowing the algorithm to locate nearest neighbors with fewer node inspections.

A.2. Latency Analysis

Figure 2(b) details the query latency (in milliseconds) across different recall levels. While QPS reflects throughput, latency is critical for online services. The results show that MCGI maintains lower latency, particularly in the high-recall regime. The logarithmic scale highlights that for difficult queries, MCGI significantly reduces the computational overhead by minimizing unnecessary disk I/O.



(a) **Sensitivity:** Recall vs. List Size (L)



(b) **Latency:** Latency vs. Recall (Log Scale)

Figure 2. **Supplementary Analysis on GIST1M.** (a) MCGI reaches high recall with a smaller search budget (L). (b) MCGI achieves lower latency for the same recall target, validating its efficiency.