Stitching Inner Product and Euclidean Metrics for Topology-aware Maximum Inner Product Search

Tingyang Chen
Zhejiang University
Hangzhou, China
chenty@zju.edu.cn

Cong Fu
Shopee Pte. Ltd.
Singapore, Singapore
fc731097343@gmail.com

Xiangyu Ke*
Zhejiang University
Hangzhou, China
xiangyu.ke@zju.edu.cn

Yunjun Gao
Zhejiang University
Hangzhou, China
gaoyj@zju.edu.cn

Yabo Ni
Nanyang Technological University
Singapore, Singapore
yabo001@e.ntu.edu.sg

Anxiang Zeng
Nanyang Technological University
Singapore, Singapore
zeng0118@ntu.edu.sg

Abstract

Maximum Inner Product Search (MIPS) is a fundamental challenge in machine learning and information retrieval, particularly in high-dimensional data applications. Existing approaches to MIPS either rely solely on Inner Product (IP) similarity, which faces issues with local optima and redundant computations, or reduce the MIPS problem to the Nearest Neighbor Search under the Euclidean metric via space projection, leading to topology destruction and information loss. Despite the divergence of the two paradigms, we argue that there is no inherent binary opposition between IP and Euclidean metrics. By stitching IP and Euclidean in the indexing and search algorithms design, we can significantly enhance MIPS performance. Specifically, this paper explores the theoretical and empirical connections between these two metrics from the MIPS perspective. Our investigation, grounded in graph-based search, reveals that different indexing and search strategies offer distinct advantages for MIPS, depending on the underlying data topology. Building on these insights, we introduce a novel graph-based index called Metric-Amphibious Graph (MAG) and a corresponding search algorithm, Adaptive Navigation with Metric Switch (ANMS). To facilitate parameter tuning for optimal performance, we identify three statistical indicators that capture essential data topology properties and correlate strongly with parameter tuning. Extensive experiments on 12 real-world datasets demonstrate that MAG outperforms existing state-of-the-art methods, achieving up to 4× search speedup while maintaining adaptability and scalability.

CCS Concepts

• **Information systems → Specialized information retrieval**; *Retrieval models and ranking*; Database query processing.

*Corresponding author.

SIGIR '25, Padua, Italy

ACM ISBN 979-8-4007-1592-1/2025/07
https://doi.org/10.1145/3726302.3730088

Keywords

Maximum inner product search;High dimensional;Proximity graph.

ACM Reference Format:
Tingyang Chen, Cong Fu, Xiangyu Ke, Yunjun Gao, Yabo Ni, and Anxiang Zeng. 2025. Stitching Inner Product and Euclidean Metrics for Topology-aware Maximum Inner Product Search. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3726302.3730088

1 Introduction

Maximum Inner Product Search (MIPS) is a fundamental task in machine learning and information retrieval [21, 33], particularly with the widespread use of high-dimensional vector representations based on inner product or cosine similarity. Key applications, including recommendation systems [45], query-answering chatbots [6], multi-modal retrieval [41], and Retrieval Augmented Generation (RAG) [7], rely on efficiently searching large vector databases to identify items that maximize similarity to a query vector. Fast and accurate MIPS enhances system performance and user experience by leaving more time for complex model inference.

Two primary paradigms for fast MIPS have emerged. The first directly operates in the inner product space [18, 19, 22, 26, 48], constructing specialized indices. However, the absence of triangle inequality *undermines the geometric theoretical support* of efficient indexing, resulting in drawbacks like high memory cost, excessive computations, and susceptibility to local optima [38]. The second paradigm [34, 48, 49] reduces MIPS to the nearest neighbor search (NNS) problem in a transformed Euclidean space, enabling the use of advanced NNS indices. However, this reduction often *relies on nonlinear projections and strong theoretical assumptions* [48, 49], which can distort data topology and cause information loss [26], leading to reduced performance and scalability.

These paradigms differ fundamentally in their reliance on distinct distance metrics. However, we argue that *Inner Product and Euclidean metrics are not mutually exclusive*. Focusing on graph-based retrieval [42], our theoretical and empirical analysis (§3) reveals that both Euclidean- and IP-based strategies can enhance MIPS efficiency. Specifically, we observe that: **(1)** Euclidean-based indexing and search ensure **strong connectivity and global reachability**, as established in prior studies [17], but typically require **longer traversal paths** to reach relevant candidates in MIPS. **(2)**

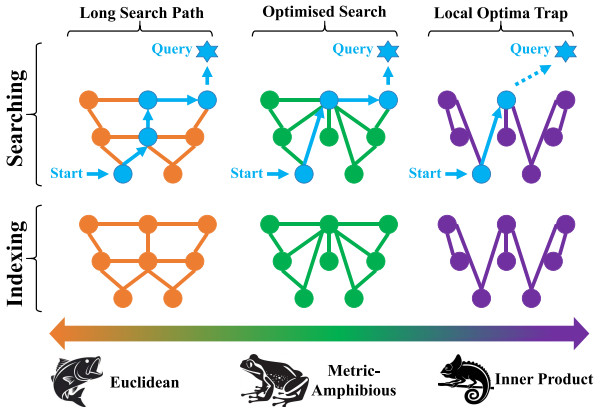

Figure 1: Illustration of our motivation. Euclidean-based indexing and searching ensure strong connectivity but suffer from inefficient traversal. IP-based indexing and search accelerates navigation toward high-relevance regions but risks getting trapped in local optima. Integrating both strategies balances connectivity, search speed, and adaptability to diverse data topologies, leading to improved performance.

IP-based indexing and search **concentrate edges toward high-norm points**, accelerating retrieval among high-IP candidates [22] but suffer from reduced connectivity and local optima traps due to in-degree concentration. In addition, the absence of an effective edge sparsification strategy with theoretical guarantees results in high memory costs and excessive computations.

To leverage the strengths of both paradigms while mitigating their limitations, we propose a hybrid framework that integrates Euclidean- and IP-based strategies for indexing and search. In the indexing phase, we introduce a novel IP-based edge selection strategy, which is combined with the Euclidean-based approach to balance global connectivity with rapid convergence to target items. In the search phase, we dynamically adjust the navigation between Euclidean- and IP-based traversal to optimize robustness and avoid local optima traps. Figure 1 illustrates our design and motivation.

Contributions. Our contributions are highlighted as follows:

(1) Theoretical Foundations: We establish comprehensive connections between IP and Euclidean metrics in the MIPS setting and introduce IP-oriented edge selection strategies with theoretical guarantees for improved efficiency.

(2) Novel MIPS Framework: We propose Metric-Amphibious Graph (MAG), a new framework that integrates Euclidean- and IP-based edges, along with a novel search algorithm, namely Adaptive Navigation with Metric Switch (ANMS). Notably, our approach **DOES NOT** require space transformation, avoiding related drawbacks that plague prior methods.

(3) Data-Driven Parameter Tuning: We introduce three statistical indicators that capture key data topology properties, providing insights for efficient parameter tuning. These indicators enable the seamless adaption of MAG to various data distributions.

(4) Comprehensive Experiments: We evaluate our approach on 12 real-world datasets with varying topology, cardinality, dimensionality, and modalities. Our approach achieves up to 4× speedup compared to advanced graph-based retrieval methods.

2 Preliminaries

Notations. Let $\mathbb{R}^d$ denote d-dimensional real coordinate space. $\{\cdot\}$ denote sets. $\mathcal{D} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ denote a vector dataset. $\langle x, y\rangle$ denotes the inner product (IP) between vector x and y. $\|x\|$ gives the Euclidean norm of vector x. V_x denotes the Voronoi Cell under IP metric associated with x. $G = (V, E)$ denotes a graph, where V is the node set and E is the edge set. $\sup(S)$ denotes the supremum of a set S. $O(\cdot)$ is the big O notation.

Problem Definition. The **Maximum Inner Product Search (MIPS)** problem is defined as: Given a query vector $q \in \mathbb{R}^d$ and a dataset $\mathcal{D}$, find the vector x^* such that $x^* = \arg\max_{x\in\mathcal{D}}\langle q, x\rangle$. Similarly, the **Nearest Neighbor Search (NNS)** problem in Euclidean space is defined as finding the vector x^* such that $x^* = \arg\min_{x\in\mathcal{D}}\|q - x\|$. Recently, researchers have focused on the **approximate** MIPS/NNS problem, which allows for an acceptable loss in accuracy for faster query processing. The approximate MIPS problem is defined as follows: Given a query $q \in \mathbb{R}^d$, a dataset $\mathcal{D} \subset \mathbb{R}^d$, and an approximation ratio $\epsilon \in (0, 1)$, let $x^* \in \mathcal{D}$ be the exact MIPS solution for q. The goal is to find a vector $x \in \mathcal{D}$ satisfying: $\langle x, q\rangle \geq \epsilon \cdot \langle x^*, q\rangle$.

Graph-based Indexing. Graph-based indices have gained prominence in NNS due to their efficiency in Euclidean spaces [17, 24, 42]. Similarly, MIPS-oriented graph-based methods often construct indices based on the IP-Delaunay Graph, analogous to the Delaunay Graph in Euclidean space [26]. IP-Delaunay Graph is defined as:

Definition 1 (**IP-Delaunay Graph**). *Given a dataset $\mathcal{D} \subset \mathbb{R}^d$, the **IP-Voronoi Cell** associated with a vector $x \in \mathcal{D}$ is $V_x = \{y \in \mathbb{R}^d \mid \langle y, x\rangle > \langle y, z\rangle, \forall z \in \mathcal{D}, z \neq x\}$. the IP-Delaunay Graph G is constructed by connecting any two nodes x_i and x_j with a bi-directional edge if their Voronoi cells V_{x_i} and V_{x_j} are adjacent in $\mathbb{R}^d$.*

The IP-Delaunay Graph is inherently densely connected, especially in high dimensions, which is inefficient for MIPS. However, unlike in Euclidean space, there lacks an efficient way to sparsify the IP-Delaunay graph to improve search efficiency and lower memory requirements, while maintaining theoretical guarantees [26].

Graph-based Search. Graph-based search follows a common iterative routine across both IP and Euclidean metrics: (1) Start from an initial candidate set; (2) Expand search iteratively by checking neighboring nodes; (3) Update candidate pool based on proximity to the query; (4) Continue until convergence criteria are met (see Algorithm 1). The key difference lies in the search objective. Euclidean Search minimizes $\|x - q\|$, while MIPS maximizes $\langle x, q\rangle$.

3 Theoretical & Empirical Analysis

Graph-based search efficiency is often analyzed using the formula $C = D \times L$, where C represents the total computation, D is the average out-degree, and L is the search-path length [42]. Optimizing efficiency requires minimizing both D and L, with an underlying assumption on strong graph connectivity. In the Euclidean space, many studies [17, 24, 28] have explored edge-sparsification strategies for Delaunay Graphs, providing theoretical guarantees on reachability and short search-path lengths. While in MIPS context, three fundamental questions arise: **(1)** How to design a sparse graph

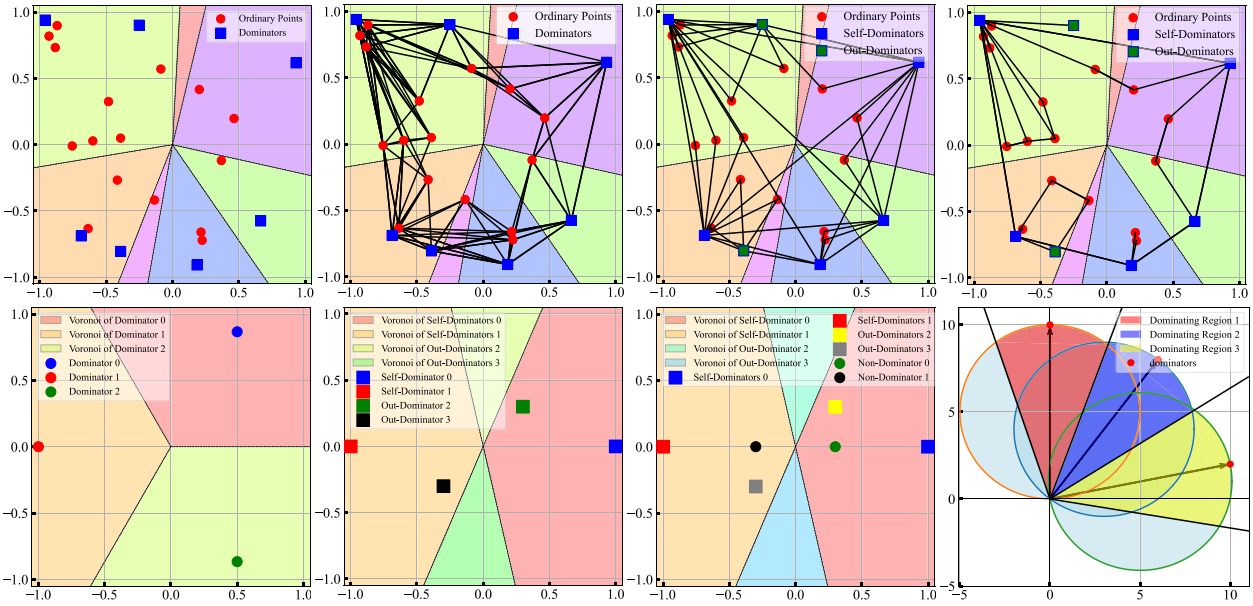

Figure 2: Illustrations of IP geometry concepts, simulated on small toy 2D data. (a) IP-Voronoi cells (open hyper-cones) with associated dominators. (b) K-Maximum Inner product (K-MIP) Graph . (c) K-Naive Dominator Graph (NDG). (d) Optimal MAG. (e) Simpler illustration of self-dominators. (f) Showcase of out-dominators dominating vacant regions. (g) Showcase of ordinary points residing in self-dominators' Voronoi$_{ip}$ cells. (h) Valid dominating region of dominators—capped hyper-cones.

Algorithm 1: Greedy Search For Graphs

Data: Graph G, query q, candidate set size l_s, result set size k, similarity measure $M(\cdot,\cdot)$.
Result: Top-k result set R.
Initialize candidate set Q with size l_s randomly;
Configure Q as heap prioritizing points closest to q w.r.t M;
while *There exists unvisited points in* Q **do**
  $p \leftarrow$ first unvisited point in Q; Mark p as visited;
  $N_p \leftarrow$ neighbors of p in G;
  for *each node* n *in* N_p **do**
    $Q.insert((n, M(n,q)))$;
  $Q.resize(l_s)$;
return $R \leftarrow$ Top-k points in Q;

structure under IP metric? **(2)** How to ensure the reachability of MIPS solutions? **(3)** What are the expected search path lengths?

To address these issues and motivate this work, we propose and analyze a geometry-based domination property under the IP metric, explore the connections between Euclidean-based NNS and MIPS, and provide empirical observations to validate our theoretical analysis. These insights inform the design of our proposed solutions, which are detailed in §4.

3.1 Geometry Domination under Inner Product

We begin by introducing dominators, tailored to the MIPS problem as a key to effective graph sparsification.

Definition 2 (**Dominators**). *A vector $x \in \mathcal{D}$ is a dominator of its Voronoi$_{ip}$ cell V_x if, for all $y \in V_x$ and all $z \in \mathcal{D}$ with $z \neq x$, it holds that $\langle y, x\rangle > \langle y, z\rangle$. Let S_{dom} represent dominators in $\mathcal{D}$.*

Intuitively, if a query vector q lies in Voronoi cell V_x dominated by point x, then x is the exact MIPS answer for q. This observation motivates the construction of the Naive Dominator Graph below:

Definition 3 (**Naive Dominator Graph (NDG)**). *Given a dataset $\mathcal{D} \subset \mathbb{R}^d$, an **NDG** G is constructed as follows: For each point $x_i \in \mathcal{D}$, sort the remaining points in descending order of $\langle x_i, y_j\rangle$ to form a list $L(x_i)$. Starting from the beginning of $L(x_i)$, evaluate each point y_j using these conditions: **(1):** $\langle y_j, y_j\rangle \geq \langle y_j, y_k\rangle$ for all $k < j$. **(2)** $\langle y_k, y_k\rangle \geq \langle y_j, y_k\rangle$ for all $1 < k < j$. If y_j satisfies these conditions, add a bi-directional edge (x_i, y_j) to the graph G.*

Under this construction, two types of dominators emerge (illustrated in Figure 2): **(1)** x is a **self-dominator** if $\forall y \in \mathcal{D}$, $\langle x, x\rangle > \langle x, y\rangle$, meaning x dominates itself and resides within V_x; **(2)** x is an **out-dominator** if x is a dominator but $\exists y \in \mathcal{D}$ such that $\langle x, x\rangle \leq \langle x, y\rangle$, then x is dominated by y and belongs to V_y. We can prove the following properties for NDG:

Theorem 1. *Given a dataset $\mathcal{D} \subset \mathbb{R}^d$, (1) an NDG is a strongly connected graph; (2) $\forall x_i \in \mathcal{D}$, x_i is connected to at most one out-dominator and all self-dominators in an NDG.*

Proof. Consider a point $x_i \in \mathcal{D}$ and the sorted list $L(x_i) = [y_1, y_2, \ldots, y_m]$ that satisfies Definition 3, we then have $\langle y_1, x_i\rangle > \langle y_j, x_i\rangle, \forall j > 1$. Therefore, y_1 dominates x_i but is not ensured to be a self-dominator, i.e., a potential out-dominator.

For any remaining point $y_j, \forall j > 1$, Definition 3 ensures that: **(1)** For any pair of y_j and y_k with $k < j$, we have $\langle y_j, y_j\rangle \geq \langle y_j, y_k\rangle$. **(2)** For any pair of y_j and y_l with $l > j$, we have $\langle y_j, y_j\rangle \geq \langle y_j, y_l\rangle$.

This confirms that $\forall j > 1, y_j$ is a self-dominator, while all the other filtered points cannot be self-dominators. Consequently, each

x_i is linked to at most one out-dominator and all self-dominators, proving Property (2). Since all edges are bi-directional, and each node is connected to at least one self-dominator, the graph is strongly connected, proving Property (1). □

By Property (2) in theorem 1, NDG's sparsity is determined by the number of self-dominators, which can be estimated analytically:

Proposition 1. *Given a dataset $\mathcal{D} \subset \mathbb{R}^d$ where vectors are element-wise i.i.d. and drawn from the standard Gaussian distribution $\mathcal{N}(0,1)$, the probability that a vector $x \in \mathcal{D}$ with norm $\|x\| = r$ is a self-dominator is given by $\mathcal{P}_{dom}(x) = \Phi(r)$, where $\Phi(\cdot)$ is the cumulative distribution function (CDF) of the standard Gaussian.*

Proof. By definition, x of norm r is a self-dominator with the probability: $\mathcal{P}_{dom}(x) = P(\langle x, x\rangle > \langle x, y\rangle \mid \|x\| = r)$.

Given $\|x\| = r$, $\langle x, x\rangle = r^2$ is deterministic. Since y is independent of x and its elements are i.i.d., conditioned on $\|x\| = r$, the inner product $\langle x, y\rangle$ becomes a linear combination of d independent standard Gaussian and follows: $\langle x, y\rangle \mid \|x\| = r \sim \mathcal{N}(0, r^2)$

By plugging in and rearranging, equation (3.1) becomes:

$$\mathcal{P}_{dom}(x) = P\left(\frac{\langle x,y\rangle}{r} < r \mid \|x\| = r\right)$$

Hence by standardizing, $Z = \frac{\langle x,y\rangle}{r} \sim \mathcal{N}(0,1)$, we have:

$$\mathcal{P}_{dom}(x) = P(Z < r) = \Phi(r)$$

where $\Phi(\cdot)$ is the CDF of the standard Gaussian distribution. □

Proposition 1 confirms that **self-dominators are predominantly high-norm vectors**, consistent with prior empirical findings [22]. Specifically, if a vector x satisfies $||x|| > 4$, it is almost sure to be a self-dominator (note that "4" is derived from the $\mathcal{N}(0,1)$ assumption and may shift depending on the actual distribution's mean and variance). Given that $||x||$ follows a Chi distribution under the assumptions in Proposition 1, the expected number of self-dominators in a dataset $\mathcal{D}$ can be estimated as: $n \times P(\|x\| > r) = n\left(1 - \frac{\gamma(d/2, r^2/2)}{\Gamma(d/2)}\right)$, where $\gamma(\cdot)$ is the lower incomplete gamma function and $\Gamma(\cdot)$ is the gamma function. This expression shows that *dominator density is also related to the dimensionality d*. Notably, d should be considered as the intrinsic dimensionality of the data, which captures the effective degrees of freedom in the data and can be significantly lower than the actual dimensionality. This, in turn, affects the expected norm distribution and the density of self-dominators. For example, in the 784-dimensional MNIST1M dataset (Table 2), only 6.2% of vectors are self-dominators, highlighting high NDG sparsity in certain structured data (refer to §4).

Despite the general sparsity of self-dominators, the NDG can exhibit dense connectivity in certain data distributions. Analogous to K-MIP graphs, which approximate IP-Delaunay graphs as a practical alternative (avoiding high-degree and enhancing memory efficiency [26]), we propose **K-NDG** as an efficient approximation of NDG: K-NDG links each node only to dominators that maximize the inner product. Unlike K-MIP, which connects each node to any high-IP neighbors, K-NDG restricts edges to dominators, leading to a sparser and more memory-efficient structure (see Figure 2).

Remark 3.1. Above theoretical advancements pioneers in the literature of MIPS and can be summarized as: **(1)** Dominators are optimal MIPS answers; **(2)** All dominators are reachable in NDG; **(3)** K-NDG is a sparser and practical alternative for IP-Delaunay.

Table 1: The performance improvement of K-NDG relative to the best competitor for various K values at 98% recall.

Datasets	k=1	k=20	k=50	k=100
Music100	38%	30%	27%	21%
Shopee1M	100%	-20%	Precision limit (0.68)	Precision limit (0.49)

Despite these merits, two structural challenges remain: **(1)** The norm distribution bias and sparsity of self-dominators lead to a high concentration of out-edges toward them, increasing the risk of local optima traps and inherent low-connectivity of K-NDG. **(2)** For top-K retrieval tasks, where not all solutions are dominators, K-NDG may struggle with reduced generalizability.

Our preliminary tests uncover these concerns. Specifically, we conduct top-K MIPS experiments on two datasets, varying K in $[1, 100]$. Table 1 presents the results. **Key observations** are: (1) As K increases, K-NDG's relative speedup over the best competitor decreases significantly. (2) On Shopee1M, K-NDG underperforms the baseline as K grows. At certain K, it even hits an accuracy bottleneck. This confirms our analysis that K-NDG may suffer from connectivity issues, potentially trapping search in local optima.

To tackle these issues, the next section investigates connections between NNS and MIPS **without requiring space transformations** to utilize the strengths of Euclidean-based methods for MIPS.

3.2 Connect Euclidean NNS To MIPS on Graphs

3.2.1 Euclidean-Based Graphs Strengthens Connectivity For MIPS. Existing nearest neighbor graphs (NNGs) such as HNSW [24] and NSG [17] ensure strong connectivity, retain high sparsity, and have been extensively tested at scale. Such advantages can benefit MIPS on graphs with proper utilization. Prior work [13] establishes a fundamental duality between MIPS and NNS, formalized as follows:

Fact 1. *Given vector database $\mathcal{D} \subset \mathbb{R}^d$ and Euclidean proximity graph $G = (V, E)$, for any $q \in \mathbb{R}^d$, there exists a scalar $\bar{\mu}$ such that for all $\mu > \max(\bar{\mu}, 0)$, the nearest neighbor of $q' = \mu q$ in $\mathcal{D}$ aligns with the* MIPS *solution for q. Furthermore, when using the standard Graph Nearest Neighbor Search (*GNNS*) on G [29], the search behavior for q under the* IP *metric matches that for q' under the Euclidean metric.*

This result implies that greedy search (Algorithm 1) under IP metric can be performed directly on NNGs, making them potentially effective indices for MIPS. From our perspective, this suggests that **edges selected under the Euclidean metric can complement** IP**-oriented graphs, enhancing connectivity, particularly for top-K retrieval**, where solutions may not be dominators.

3.2.2 Euclidean Oriented Navigation Avoids Local Optima in MIPS. The MIPS objective can be stated as: $\max\langle p, q\rangle = \max\|p\|\|q\| \cos\theta$. Thus, MIPS consists of two key processes: norm expansion of $\|p\|$ and minimizing angle θ. We now show that executing a Euclidean-oriented search on a query q effectively reduces angular distance.

Proposition 2. *Consider a vector database $\mathcal{D} \subset \mathbb{R}^d$ containing n points, where n is sufficiently large to ensure robust statistical properties. Assume each element of the base vectors are sampled from $\mathcal{N}(0,1)$, independently and identically distributed (i.i.d.). The angle θ between point x and its nearest neighbor y in $\mathcal{D}$ can be estimated*

by $\arccos\min\left(\left(\frac{1}{td}\left(\log n+\frac{d}{2}\log(1-t^2)^{-1}\right)\right),1\right)$, *where* $0<t<1$ *is a tuning parameter. As* $n\to\infty$, θ *converge to* 0° .

Proof. Because y is the nearest neighbor of x, we have $y = \arg\min_z ||x-z||$. Given $||x-z|| = \sqrt{||x||^2+||z||^2-2\langle x,z\rangle}$, we can get $y = \arg\max_z\langle x,z\rangle$ subject to $2\langle x,z\rangle < ||x||^2+||z||^2$.

Given x are i.i.d. sampled from $\mathcal{N}(0,1)$, $||x||^2$ follows a chi-square distribution with d degree of freedom. According to Central Limit Theory and concentration of measure, $||x||^2$ distribution will concentrate sharply around d, making d a good estimation of $\|x\|$. By Substituting, we can get $y\approx\arg\max_y\langle x,y\rangle, s.t.\langle x,y\rangle<d$.

Let $M=\max\langle x,y\rangle$, which follows a distribution characterized by a modified Bessel function of the second kind [10], given x,y are i.i.d sampled from $\mathcal{N}(0,1)$. Using Extreme Value Theory, we can estimate $\max\langle x,y\rangle$ below. With Jensen's Inequality [25], we have:

$$e^{t\mathbb{E}[M]}\le E[e^{tM}]\le nMGF(t)_M$$
$$\mathbb{E}[M]\le\frac{1}{t}\left(\log n+\frac{d}{2}\log(1-t)^{-1}\right),$$

where $MGF(t)_M=(1-t^2)^{-d/2}$ is the Moment Generating Function (MGF) [14] of distribution M, and $0<t<1$ is a tuning parameter for the tightness of estimation. According to Extreme Value Theory, $\mathbb{E}[M]$ can be estimated by $\frac{1}{t}\left(\log n+\frac{d}{2}\log(1-t)^{-1}\right)$. Given $\cos\theta=\frac{\langle x,y\rangle}{||x||||y||}$, θ can be estimated as $\arccos\left(\frac{1}{td}\left(\log n+\frac{d}{2}\log(1-t)^{-1}\right)\right)$. Given that $\langle x,y\rangle<d$, we can then approximate that $\cos\theta\approx\min\left(\frac{1}{td}\left(\log n+\frac{d}{2}\log(1-t)^{-1}\right),1\right)$. When n grows to ∞, $\cos\theta$ converges to 1 and θ converges to 0°. □

Remark 3.2-Key takeaways from the above analysis: (1) Section 3.2.1 shows Euclidean-based edge selection can effectively address the connectivity limitations of K-NDG, particularly for top-K retrieval. **(2) Section 3.2.2** demonstrates that Euclidean-oriented search inherently reduces angular distance, which aligns with one of the core objectives of MIPS.

While Euclidean-oriented edges provide global connectivity and potential reachability to MIPS solutions, a critical argument is that **executing** MIPS **solely on Euclidean-oriented graphs can be inefficient and prone to local optima traps**.

We conducted another preliminary test to evaluate MIPS on NNGs (Figure 3). Specifically, we run top-100 MIPS on real-world datasets using NSG [17]. Findings are: (1) NNGs achieve competitive recall (e.g., 98% average recall on the Imagenet-1k dataset). (2) Certain queries receive 0 recall, leading to a recall bottleneck.

Upon analyzing the failure cases on the Imagenet-1K dataset, we identified that these failures stem from suboptimal navigation: Without proper guidance, the search process maximizes the inner product inefficiently by alternating between norm expansion and angular minimization. This unordered navigation leads the search into local optima traps, as illustrated in the upper row of Figure 3.

Given the above analyses, we then tested the following **Metric-Amphibious strategies**: (1) Add r edges from K-NDG for each node in NSG, linking them to the closet dominators; (2) Replace the first m steps of MIPS navigation with Euclidean-oriented navigation, prioritizing angular minimization within the candidate set Q (Algorithm 1). This Metric-Amphibious approach significantly

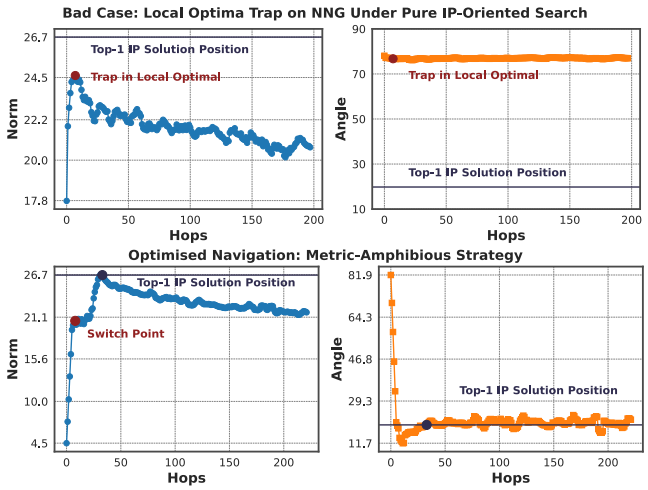

Figure 3: MIPS processes on Imagenet-1K Dataset through the lens of the average norm of the candidates, and the angles between the candidates and the query. The upper row illustrates a failure case when executing MIPS directly on an NSG, where the algorithm fails to locate solutions even after 200 iterations. The lower row depicts the optimized search process using Metric-Amphibious strategies, which successfully identifies the top-1 solution within 50 iterations and achieves 100% recall within 200 iterations.

improves recall, raising previously failed queries (0 recall) to 99% recall within the same number of iterations. The lower row of Figure 3 demonstrates that, for the same query, the new strategy finds the top-1 solution within 50 iterations and locates 100% of top-100 solutions within 200 iterations. In contrast, the upper bad case fails to find any solution even beyond 200 iterations.

4 Methodology

The theoretical and empirical insights from Section 3 highlight the potential of Metric-Amphibious Indexing and Search. Specifically: (1) Combining Euclidean- and IP-based edge selection enhances graph connectivity and improves robustness in top-K retrieval; (2) Integrating Euclidean- and IP-based search navigation reduces the risk of search trapping in local optima. Motivated by these findings, we propose a novel indexing and search framework below.

4.1 Metric-Amphibious Graph

Definition 4 (MAG). *Given an MRNG [17] G constructed on a dataset $\mathcal{D}\subset R^d$, we extend it by identifying for each point $x\in\mathcal{D}$ r dominator neighbors q that maximizes $\langle x,q\rangle$, using the strategy in Definition 3. The resulting graph G' is referred to as the Metric-Amphibious Graph (MAG).*

Theorem 2. *An* MAG *G defined on $\mathcal{D}\subset\mathbb{R}^d$ has the following properties: (1) G is strongly connected; (2) the* MIPS *answer x^* of query q is reachable from any starting point $x\neq x^*$ via a greedy search under the* IP *metric (Algorithm 1); the amortized search complexity of* MIPS *on G is $O(\frac{cn^{1/d}\log(n)}{\Delta(n)})$, where $n=|\mathcal{D}|$, c is a constant, and $\Delta(n)$ is a very slowly decreasing function of n [17].*

Proof. By Theorem 3 in [17], MRNG is strongly connected, ensuring at least one path between any node pair. The extended edges per node does not alter this connectivity, thus proving property (1).

By Theorem 1 in [17], MRNG guarantees a path between any two nodes can be found via Algorithm 1 under the Euclidean metric. Adding extra edges in MAG does not break this property. Moreover, by Fact 1, the MIPS solution x^* for query q is reachable under the Euclidean metric by scaling q to a hypothesis query μq. Due to this duality between MIPS and NNS, this path can also be found guided by the IP metric with Algorithm 1, proving Property (2).

Following the same routine of proving Theorem 3 in [17], we can prove MAG is an MSNET. By Theorem 2 in [17], the expected search path length in an MSNET follows: $\mathbb{E}[L_{path}] = \frac{n^{1/d}\log(n)}{\Delta(n)}$. Since MAG's out-degree is at most $R + r$ (where R is the MRNG's maximal out-degree and can be treated as a constant given a fixed dimension d [17]), the amortized search complexity of MIPS on G is $O(\frac{cn^{1/d}\log(n)}{\Delta(n)})$, where c absorbs $d(R + r)$, proving Property (3). □

Key Takeaways: By Theorem 2, MAG retains the connectivity strength of MRNG and benefits from the shortcuts to dominators from K-NDG, **enhancing navigation efficiency without affecting graph sparsity and without space transformation**.

Despite the theoretical advantages, a key limitation is the indexing complexity and memory efficiency. The standard construction of MAG incurs $O(DN^2)$ complexity, making it impractical for large datasets. Additionally, the high variance in out-degree among nodes may lead to memory inefficiencies. To address this, we propose a scalable approximation: (1) Construct MAG using approximate K-NN graphs (as an approximation for the Euclidean Delaunay graph); (2) Restrict the number of MRNG- and NDG-based edges, balancing efficiency and connectivity, detailed as follows.

4.1.1 Two-Stage Construction Algorithm of MAG. By Fact 1, the K-MIP solutions for a query q can be derived from a Euclidean proximity graph. However, obtaining q's Euclidean neighbors from an IP-oriented graph is inefficient due to potential low connectivity (Section 3.1). Based on this, we propose the following pipeline.

Stage 1: K-MRNG Approximation. According to [17], an approximate MRNG can be constructed from a K-NNG. For each point $x \in \mathcal{D}$, we use IVF-PQ [4] to retrieve its K nearest Euclidean neighbors, forming a K-NNG. Then we use MRNG's edge selection strategy [17] to sparsify this K-NNG. Finally, we retain only the closest K_1 neighbors, forming the approximate K-MRNG.

Stage 2: K-MAG Approximation. Utilizing Fact 1, we execute Algorithm 1 using IP metric to obtain K-MIP neighbors for each point $x \in \mathcal{D}$ on above K-MRNG. Then we apply NDG's edge selection strategy (Definition 3) to sparsify the K-MIP neighbors. Finally, we retain only the closest K_2 IP-oriented neighbors and inject them into K-MRNG, forming the approximate K-MAG.

4.1.2 Metric-Amphibious Index Loading. To improve adaptability and memory efficiency, not all $K_1 + K_2$ edges are used during the search phase. Instead, we leverage the parameter R to control the maximum out-degree of the Metric-Amphibious Graph (MAG) and the parameter $\alpha \in (0, 1)$ to control the proportion of IP-oriented edges. Specifically, αR IP-oriented edges and $(1 - \alpha)R$ Euclidean-oriented edges are dynamically loaded, optimizing performance for different data distributions and varying K in top-K retrieval.

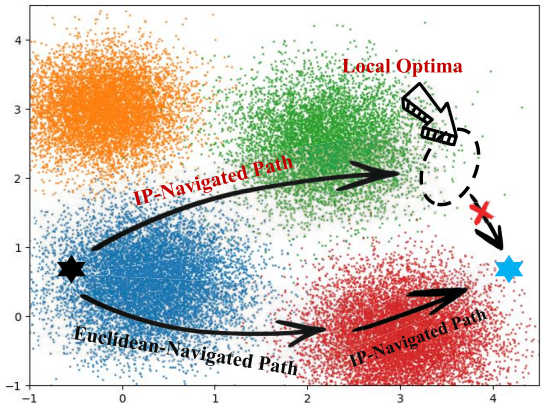

Figure 4: Illustration of the impact of clustering. Without proper constraints, MIPS tends to scale the norm or minimize the angle randomly during IP maximization, depending on the data distribution and starting point. This process often traps the search in large-norm, sparsely populated regions (local optima). By incorporating Euclidean-oriented edges and improved navigation, this issue can be mitigated.

Indexing complexity. The overall complexity consists of two parts: (1) Build a K-MRNG and (2) inject NDG edges. Let n denote the number of vectors in $\mathcal{D}$, R the maximum out-degree, and d the dimension. The time complexity for the first step has an empirical complexity $O(ndR \log n)$ [17]. The second step involves two phases, where the search phrase has a complexity of $O(c_1 n^{1/d} \log n)$ [17] and the pruning incurs $O(c_2 d)$ per point. c_1, c_2 are constants. Overall, $O(ndR \log n)$ dominates the indexing time complexity.

4.2 Adaptive Navigation with Metric Switch

Our analysis in Section 3 highlights the benefits of combining different navigation strategies to prevent the search from getting trapped in local optima when maximizing IP. This requires a minor modification to Algorithm 1. Formally, we propose **Adaptive Navigation with Metric Switch (ANMS)**, a two-stage search strategy that dynamically switches from Euclidean to IP-based navigation. **The first stage** executes Algorithm 1 under Euclidean metric for m steps. This minimizes the angular distances between the candidates and the query q; **The second stage** resumes Algorithm 1 upon the resulting candidates while switching the metric to maximize IP. Here, m is an extra tuning parameter to adapt ANMS to various data distributions, providing flexibility in controlling the initial search direction before switching metrics.

The search complexity of the ideal MAG is derived in Theorem 2. While the approximate construction of MAG may slightly compromise this theoretical efficiency, our empirical evaluations in Section 5 (Figure 7) demonstrate that the search process still scales nearly as $O(\log n)$. These results confirm that the approximation introduced during construction has a minimal impact on the overall performance, ensuring that the search remains efficient.

4.3 Distribution Aware Parameter Tuning

In Section 4.1 and 4.2, integrating Euclidean- and IP-based strategies introduces two balancing parameters. This raises two important

questions: **(1) How to tune the proportion (α) of Euclidean- and IP-based edges? (2) How to determine the metric switch position (m) in ANMS?** We find that these parameters are closely tied to the data distribution and retrieval quantity K, allowing us to derive key statistical indicators for efficient parameter tuning.

4.3.1 High Sparsity of Dominators Favors IP-Oriented Tuning. When dominators are sparse, **IP-Oriented tuning** should be prioritized. This is because MIPS solutions tend to concentrate in a smaller subset of the dataset, necessitating: **(1) Increase** α, the fraction of IP-oriented edges, to enhance direct connectivity to dominators; **(2) Reduce** m, the number of Euclidean-based search steps, to accelerate convergence to MIPS solutions. Otherwise, the data favors Euclidean-oriented tuning, i.e., reducing α and increasing m.

As stated in Proposition 1, the proportion of dominators in a dataset is highly dependent on the norm distribution of the vectors. Specifically, flatter norm distributions (with greater variability in norms) result in fewer dominators. In comparison, sharper norm distributions lead to a higher dominator proportion as it is harder for one point to dominate another with a similar norm. Another interpretation is that datasets with sharply distributed norms tend to approximate spherical distributions. In such cases, MIPS approaches become closer to maximizing cosine similarity, which is inherently aligned with Euclidean-based NNS, thus favoring more Euclidean-oriented tuning.

To quantitatively guide parameter tuning, we introduce the Coefficient of Variation (CV) on the norm distribution, defined as $\sigma(\|x\|)/\mathbb{E}(\|x\|)$. Here $\sigma(\|x\|)$ is the standard deviation of vector norms. $\mathbb{E}(\|x\|)$ is the mean norm over the dataset.

Takeaways: High CV (≥ 0.1) indicates more IP-oriented tuning, while low CV implies more Euclidean-oriented tuning.

4.3.2 Highly Clustered Data Favors Euclidean-Oriented Tuning. The proposition 1 highlights the impact of dominator-oriented edges in highly clustered data, where their sparsity and concentration can lead to local optima traps, especially in inter-cluster regions. In such cases, inter-cluster connectivity depends on local Euclidean neighborhoods of points on cluster boundaries (Figure 4). **Dominator-oriented edges are typically radial and may fail to bridge clusters**, causing suboptimal search paths. This issue can be mitigated using Euclidean-oriented edge selection strategies [17, 24] and navigation [12]. Thus, highly clustered datasets require more Euclidean-oriented tuning to enhance inter-cluster reachability.

To quantify the clustering, we employ the Davies-Bouldin Index (DBI), defined as: DBI= $\frac{1}{N}\sum_{i=1}^{N}\max_{j\neq i}\frac{\sigma_i+\sigma_j}{d(c_i,c_j)}$, where N is the number of clusters, σ_i is the average intra-cluster distance for cluster i, c_i and c_j are cluster centroids, and $d(c_i, c_j)$ is the distance between cluster centers. A higher DBI indicates lower inter-cluster separability. To capture clustering characteristics specific to MIPS, we compute DBI under both Euclidean distance and cosine similarity, since Euclidean DBI captures density and spatial separation of clusters, while cosine DBI accounts for vector orientation variations, both of which are critical for MIPS optimization.

Takeaways: Highly clustered data (DBI (≤ 2) under any distance metric) prioritizes Euclidean-oriented tuning, while evenly distributed data prioritizes IP-oriented tuning.

Table 2: Dataset statistics, including the size of base and query data, dimensionality (Dim.), modality, DBI(Euclidean), DBI(Cosine), and CV [11] of norm distribution.

Dataset	Base	Dim.	Query	Modality	DBI(Euc.)	DBI(Cos.)	CV
Music100	1M	100	10,000	Audio	1.5	2.8	0.25
YFCC1M	1M	100	1,000	Multi	1.51	2.9	0.07
SIFT1M	1M	128	1,000	Image	3.26	2.6	0.001
Text2Image1M	1M	200	100,000	Multi	2.5	3.0	0.03
MNIST1M	1M	784	10,000	Image	2.7	2.8	0.18
GIST1M	1M	960	1,000	Image	6.28	3.2	0.27
OpenAI-1536	1M	1536	1,000	Text	4.1	5.3	0.0
Imagenet-1k	1.3M	1536	1,000	Image	1	1.4	0.36
Color3M	3M	282	1,000	Image	2.6	2.1	0.17
Shopee10M	10M	48	1,000	E-commerce	2.4	2.1	0.24
Text2Image10M	10M	200	100,000	Multi	3.3	3.6	0.03
Laion10M	12M	512	1,000	Multi	4.3	3.6	0.0

4.3.3 Parameter tuning w.r.t. K. As shown in Section 3.1, smaller K in top-K retrieval indicates the solutions are highly concentrated among dominators, where IP-oriented tuning is preferred. Otherwise, Euclidean-oriented tuning is prioritized.

5 Experimental Evaluation

In this section, we conduct extensive and comprehensive experiments to answer the following research questions: **RQ1:** How does MAG's search and indexing perform compared to existing methods? **RQ2:** How does MAG's metric-amphibious tuning contribute to its adaptability? **RQ3:** How does MAG behave at scale?

5.1 Experimental Setup

Datasets. We evaluate 12 real-world datasets with diverse cardinality, dimensionality, topology, and modality. Among them, **Music100** [26], **MNIST1M** [1], **Imagenet-1K** [32], **SIFT1M** [8], **GIST1M** [8], **YFCC1M** [39], and **Color3M** [2] are widely used for the MIPS problem. **OpenAI-1536** [5] is derived via the OpenAI text-embedding-3-large model on DBpedia, **Shopee10M** [16] comes from a recommender system enhanced with advanced representation learning, **Text2Image** [3] and **Laion10M** [31] provide cross-modality embeddings for cross-modal retrieval.

Competitors. We compare MAG against recent advanced methods of varying types: **(1) ip-NSW** [26], a graph-based approach utilizing an inner product navigable small world graph; **(2) ip-NSW+** [22], which enhances ip-NSW by adding a high-tier angular proximity graph; **(3) Möbius Graph** [49], which utilizes the Möbius transformation to convert MIPS to NNS; and **(4) NAPG** [38], an IP native space method that introduces a norm-adjusted proximity graph. **(5) ScaNN** [19]: a quantization method that integrates the recent state-of-the-art method **SOAR** [37] to further enhance performance. **(6) Naive MRNG**: the approximate K-MRNG constructed via Stage 1 solely as a strong baseline (Section 4.1).

Implementation. All baselines are implemented in C++. The ScaNN library is called by Python bindings. Experiments are conducted on a single machine using 48 threads for index building across all methods. For query execution, we use same number of threads to ensure a fair comparison. Each experiment is repeated three times, and the average result is reported to reduce system variability. The code is at **https://github.com/ZJU-DAILY/MAG**.

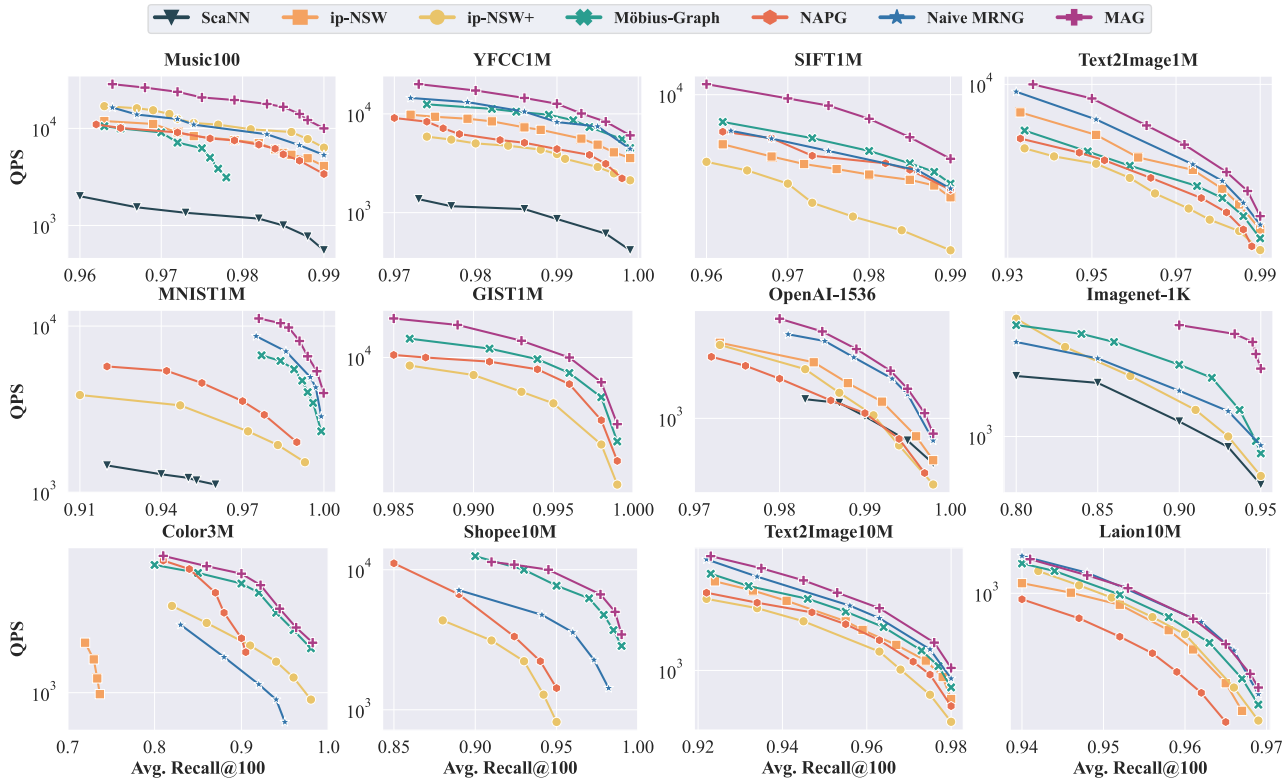

Figure 5: Experimental results of search performance on real-world datasets. The upper right is better.

Table 3: Experimental results on indexing time (s) and memory footprint (MB) of different methods on representative datasets.

	Music100		MNIST1M		GIST1M		OpenAI-1536		Imagenet-1k		Shopee10M		Text2Image10M		Laion10M	
	Time	Memory	Time	Memory	Time	Memory	Time	Memory	Time	Memory	Time	Memory	Time	Memory	Time	Memory
ScaNN	12	789	87	4666	106	5122	499	10240	1072	16820	67	7025	208	18480	1621	30656
ip-NSW	68	584	106	3256	674	3921	721	6123	175	7580	431	4464	881	9656	4590	27146
ip-NSW+	429	726	476	3588	1226	4233	1081	6206	990	7975	3217	5273	7722	11079	9728	28139
NAPG	84	648	526	3256	1086	3962	805	6123	820	7772	959	5452	1401	10250	6224	27146
Möbius Graph	95	562	138	3256	930	3921	867	6123	338	7784	469	4464	1038	9720	3862	27146
Naive MRNG	99	441	144	2960	792	3602	1334	5922	198	7505	667	2897	984	8426	2055	25882
MAG (ours)	134	496	230	2998	880	3660	1556	5990	219	7507	887	3194	1344	8795	2976	25920

Evaluation Protocol. We evaluate the query performance using the common metric **Recall vs. Queries Per Second (QPS)**, which represents the number of queries an algorithm can process per second at each specified *recall@k* level. The recall@k is defined as: $recall@k = \frac{|R \cap R'|}{|R|} = \frac{|R \cap R'|}{k}$, where R is the ground-truth set of results, and R' is the set of results returned by the algorithm. We use $k = 100$, following prior practice. The memory footprint and indexing time are reported to evaluate indexing costs.

5.2 Experimental Results

RQ1–Search. Figure 5 presents queries per second (QPS) against recall@100 for all methods. **Key findings: (1) MAG consistently outperforms all baselines** across datasets, due to its metric-amphibious framework and theoretical supports. Notably, It achieves 4× speedup over Möbius Graph on Imagenet-1K, 6× over ip-NSW+ on MNIST1M, and 1.5× over ip-NSW on Laion10M, highlighting MAG's robustness. **(2) Baselines face accuracy bottlenecks** due to weak connectivity or local optima: NAPG struggles on 8 datasets, ip-NSW stagnates on 5 datasets, Naive MRNG caps at 90% recall on GIST1M, and Möbius Graph fails at 80% recall on OpenAI-1536. **(3) Data characteristics impact:** Low CV and high DBI datasets (Laion10M and Text2Image10M) favor Euclidean tuning. Thus Naive MRNG performs well besides MAG. High CV datasets (Music100, MNIST1M) favor IP-tuning, where MAG stands out significantly. For low DBI datasets (YFCC1M, Imagenet-1K), balanced tuning gives MAG a clear advantage by efficient navigating across clusters. These align with our analysis. **(4) Dominator matters:** Despite strong connectivity, Naive MRNG underperforms as it fails to utilize concentration of MIPS solutions among dominators.

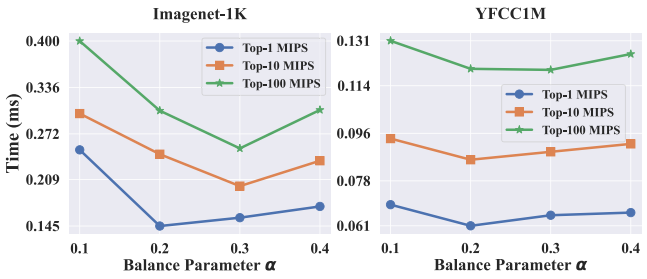

Figure 6: Search time versus different IP-edge ratio α.

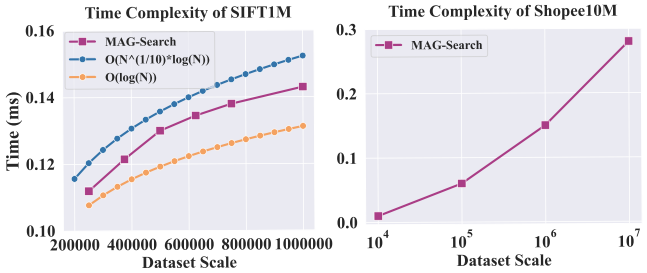

Figure 7: Search time v.s. scale on SIFT1M and Shopee10M.

RQ1–Indexing. Table 3 summarizes the indexing time and memory usage for all competitors across eight representative datasets, omitting 4 due to space limit. MAG demonstrates moderate indexing and memory costs, similar on the 4 unreported ones. While ScaNN excels in fast indexing, it sacrifices search performance significantly. Overall, MAG achieves the best search-indexing trade-off.

RQ2: Metric-Amphibious Tuning. We evaluate the impacts of Metric-Amphibious Tuning during the indexing and search phases over two representative datasets, regarding key parameters (1) the IP-oriented edge ratio α; (2) the metric switch position m.

Impact of α. We vary α, which adjusts the ratio of IP-oriented edges, and analyze its effect on search performance for top-K MIPS ($K \in \{1, 10, 100\}$) on Imagenet-1K and YFCC1M. Results in Figure 6 highlight: **(1)** Both IP- and Euclidean oriented edges contributes to the efficiency of MAG, resulting in a concave curve for search performance. **(2)** Datasets with high CV favor larger α, but excessive α reduces performance due to lower connectivity and local optima traps. Otherwise, low CV datasets favors smaller α. **(3)** Excessively large α hurts the performance under larger K more than smaller K.

Impact of metric switch position m. Fixing the indices, we vary m and evaluate its effect on search speedup over the best competitor at 99% recall. Table 4 shows: **(1)** Each dataset has a unique optimal m, driven by its specific data distribution. **(2)** Datasets with highly clustered data and sparse dominators benefit more from Euclidean-oriented navigation to avoid local optima traps.

RQ3–Scalability. We evaluate the search and indexing time complexity of MAG on SIFT1M and Shopee10M. Figure 7 shows MAG 's search complexity scales near $O(\log n)$ on Shopee10M (more IP-oriented), while slightly higher yet capped at $O(n^{1/d} \log n)$ on SIFT1M (more Euclidean-oriented). Note that the intrinsic dimension of SIFT1M is around 10. The indexing complexity of MAG follows near $O(n \log n)$ on both datasets, aligned with our analyses.

Table 4: The performance enhancement of MAG compared to the best competitor across various search switch steps on Imagenet-1k and YFCC1M.

Datasets	step=10	step=20	step=30	step=40
Imagenet-1K	1.6x	2.24x	3.7x	2.64x
YFCC1M	28%	39%	32%	25%

6 Related Works

Inner Product is crucial in AI and machine learning applications such as representation learning,language modeling, computer vision and recommender systems [7, 20, 30, 41, 44, 47]. MIPS methods are generally categorized into Locality Sensitive Hashing (LSH), tree-, quantization-, and graph-based approaches:

LSH-based methods: Traditional LSH [40, 43], originally designed for Euclidean space, is adapted for MIPS using transformations such as L_2 [34], Correlation [35], and XBOX [9]. Range-LSH [46] is the first to observe that MIPS results cluster around large-norm vectors. Simple-LSH [27] introduce a symmetric LSH that enjoys strong guarantees. Fargo [48] represents the recent state-of-the-art.

Tree-based methods: Early MIPS approaches favored trees but struggled with high dimensionality. ProMIPS [36] addresses this by projecting vectors into a lower-dimensional space, though information loss remains a challenge. LRUS-CoverTree [23] improves on this but faces difficulties with negative inner product values.

Quantization-based methods: NEQ [15] quantizes the norms of items in a dataset explicitly to reduce errors in norm. ScaNN [19] integrates "VQ-PQ" with anisotropic quantization loss, while SOAR [37] employs an orthogonality-amplified residual loss and have become state-of-the-art and been integrated into ScaNN library.

Graph-based methods: Proven effective for NNS, graph-based methods have been adapted for MIPS. ip-NSW [26] builds Delaunay graphs via inner product. ip-NSW+ [22] improves graph quality with angular proximity. Möbius-Graph [49] adopts Möbius transforms for MIPS. IPDG prunes extreme points for top-1 MIPS. NAPG [38] uses a norm-adaptive inner product ($\alpha\langle x, y\rangle$) in ip-NSW.

7 Conclusion

This paper introduces a hybrid Metric-Amphibious framework for efficient and scalable MIPS, including a novel graph index MAG and an efficient search algorithm ANMS. Comprehensive theoretical and empirical analysis support us to effectively leverage the strengths of both IP- and Euclidean-oriented strategies while mitigating their limitations. Three statistical indicators sketching the data characteristics are identified to guide efficient parameter tuning. Extensive experiments demonstrate the efficiency, adaptability, and scalability of the proposed method.

Acknowledgments

This work was supported in part by the NSFC under Grants No. (62025206 and U23A20296), Zhejiang Province's "Lingyan" R&D Project under Grant No. 2024C01259, Ningbo Yongjiang Talent Introduction Programme (2022A-237-G).

References

[1] 1998. MNIST. http://yann.lecun.com/exdb/mnist/.
[2] 2009. Color3M. http://cophir.isti.cnr.it/.
[3] 2021. Text-to-Image. https://research.yandex.com/blog/benchmarks-for-billion-scale-similarity-search.
[4] 2024. faiss. *https://github.com/facebookresearch/faiss*.
[5] 2024. OpenAI-1536. *https://huggingface.co/datasets/KShivendu/dbpedia-entities-openai-1M*.
[6] Amin Ahmad, Noah Constant, Yinfei Yang, and Daniel Cer. 2019. ReQA: An Evaluation for End-to-End Answer Retrieval Models. In *EMNLP*. 137.
[7] Akari Asai, Sewon Min, Zexuan Zhong, and Danqi Chen. 2023. Retrieval-based language models and applications. In *ACL*. 41–46.
[8] Martin Aumüller, Erik Bernhardsson, and Alexander Faithfull. 2020. ANN-Benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. *IS* 87 (2020), 101374.
[9] Yoram Bachrach, Yehuda Finkelstein, Ran Gilad-Bachrach, Liran Katzir, Noam Koenigstein, Nir Nice, and Ulrich Paquet. 2014. Speeding up the xbox recommender system using a euclidean transformation for inner-product spaces. In *RecSys*. 257–264.
[10] Frank Bowman. 1958. *Introduction to Bessel functions*.
[11] Charles E Brown. 1998. Coefficient of variation. In *Applied multivariate statistics in geohydrology and related sciences*. 155–157.
[12] Patrick Chen, Wei-Cheng Chang, Jyun-Yu Jiang, Hsiang-Fu Yu, Inderjit Dhillon, and Cho-Jui Hsieh. 2023. Finger: Fast inference for graph-based approximate nearest neighbor search. In *WWW*. 3225–3235.
[13] Tingyang Chen, Cong Fu, Kun Wang, Xiangyu Ke, Yunjun Gao, Wenchao Zhou, Yabo Ni, and Anxiang Zeng. 2025. Maximum Inner Product is Query-Scaled Nearest Neighbor. *arXiv preprint arXiv:2503.06882* (2025).
[14] John H Curtiss. 1942. A note on the theory of moment generating functions. *AMS* 13, 4 (1942), 430–433.
[15] Xinyan Dai, Xiao Yan, Kelvin KW Ng, Jiu Liu, and James Cheng. 2020. Norm-explicit quantization: Improving vector quantization for maximum inner product search. In *AAAI*. 51–58.
[16] Cong Fu, Kun Wang, Jiahua Wu, Yizhou Chen, Guangda Huzhang, Yabo Ni, Anxiang Zeng, and Zhiming Zhou. 2024. Residual Multi-Task Learner for Applied Ranking. In *SIGKDD*. 4974–4985.
[17] Cong Fu, Chao Xiang, Changxu Wang, and Deng Cai. 2019. Fast approximate nearest neighbor search with the navigating spreading-out graph. *PVLDB* 12, 5 (2019), 461–474.
[18] Ruiqi Guo, Sanjiv Kumar, Krzysztof Choromanski, and David Simcha. 2016. Quantization based fast inner product search. In *AISTATS*. 482–490.
[19] Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. 2020. Accelerating large-scale inference with anisotropic vector quantization. In *ICML*. 3887–3896.
[20] Jui-Ting Huang, Ashish Sharma, Shuying Sun, Li Xia, David Zhang, Philip Pronin, Janani Padmanabhan, Giuseppe Ottaviano, and Linjun Yang. 2020. Embedding-based retrieval in facebook search. In *SIGKDD*. 2553–2561.
[21] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *NeurIPS*. 9459–9474.
[22] Jie Liu, Xiao Yan, Xinyan Dai, Zhirong Li, James Cheng, and Ming-Chang Yang. 2020. Understanding and improving proximity graph based maximum inner product search. In *AAAI*. 139–146.
[23] Hengzhao Ma, Jianzhong Li, and Yong Zhang. 2024. Reconsidering Tree based Methods for k-Maximum Inner-Product Search: The LRUS-CoverTree. In *ICDE*. 4671–4684.
[24] Yu A Malkov and Dmitry A Yashunin. 2018. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *TPAMI* 42, 4 (2018), 824–836.
[25] EJ McShane. 1937. Jensen's inequality. *AMS* 43, 8 (1937), 521–527.
[26] Stanislav Morozov and Artem Babenko. 2018. Non-metric similarity graphs for maximum inner product search. In *NeurIPS*. 4726–4735.
[27] Behnam Neyshabur and Nathan Srebro. 2015. On symmetric and asymmetric lshs for inner product search. In *International Conference on Machine Learning*. PMLR, 1926–1934.
[28] Yun Peng, Byron Choi, Tsz Nam Chan, Jianye Yang, and Jianliang Xu. 2023. Efficient approximate nearest neighbor search in multi-dimensional databases. *SIGMOD* 1, 1 (2023), 1–27.
[29] Liudmila Prokhorenkova and Aleksandr Shekhovtsov. 2020. Graph-based nearest neighbor search: From practice to theory. In *ICML*. 7803–7813.
[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*. 8748–8763.
[31] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*. 8748–8763.
[32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *IJCV* 115 (2015), 211–252.
[33] Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *ACL*. 4430–4441.
[34] Anshumali Shrivastava and Ping Li. 2014. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). In *NIPS*. 2321–2329.
[35] Anshumali Shrivastava and Ping Li. 2015. Improved asymmetric locality sensitive hashing (ALSH) for Maximum Inner Product Search (MIPS). In *UAI*. 812–821.
[36] Yang Song, Yu Gu, Rui Zhang, and Ge Yu. 2021. ProMIPS: Efficient high-dimensional C-approximate maximum inner product search with a lightweight index. In *ICDE*. 1619–1630.
[37] Philip Sun, David Simcha, Dave Dopson, Ruiqi Guo, and Sanjiv Kumar. 2023. SOAR: improved indexing for approximate nearest neighbor search. In *NeurIPS*. 3189–3204.
[38] Shulong Tan, Zhaozhuo Xu, Weijie Zhao, Hongliang Fei, Zhixin Zhou, and Ping Li. 2021. Norm adjusted proximity graph for fast inner product retrieval. In *SIGKDD*. 1552–1560.
[39] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. 2016. Yfcc100m: The new data in multimedia research. *Commun. ACM* 59, 2 (2016), 64–73.
[40] Jingdong Wang, Ting Zhang, Nicu Sebe, Heng Tao Shen, et al. 2017. A survey on learning to hash. *TPAMI* 40, 4 (2017), 769–790.
[41] Mengzhao Wang, Xiangyu Ke, Xiaoliang Xu, Lu Chen, Yunjun Gao, Pinpin Huang, and Runkai Zhu. 2024. Must: An effective and scalable framework for multimodal search of target modality. In *ICDE*. IEEE, 4747–4759.
[42] Mengzhao Wang, Xiaoliang Xu, Qiang Yue, and Yuxiang Wang. 2021. A comprehensive survey and experimental comparison of graph-based approximate nearest neighbor search. *PVLDB* 14, 11 (2021), 1964–1978.
[43] Jiuqi Wei, Botao Peng, Xiaodong Lee, and Themis Palpanas. 2024. Det-lsh: a locality-sensitive hashing scheme with dynamic encoding tree for approximate nearest neighbor search. *arXiv preprint arXiv:2406.10938* (2024).
[44] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. 2020. Product knowledge graph embedding for e-commerce. In *WSDM*. 672–680.
[45] Jun Xu, Xiangnan He, and Hang Li. 2018. Deep learning for matching in search and recommendation. In *SIGIR*. 1365–1368.
[46] Xiao Yan, Jinfeng Li, Xinyan Dai, Hongzhi Chen, and James Cheng. 2018. Norm-ranging LSH for maximum inner product search. In *NeurIPS*. 2956–2965.
[47] Hsiang-Fu Yu, Prateek Jain, Purushottam Kar, and Inderjit Dhillon. 2014. Large-scale multi-label learning with missing labels. In *ICML*. 593–601.
[48] Xi Zhao, Bolong Zheng, Xiaomeng Yi, Xiaofan Luan, Charles Xie, Xiaofang Zhou, and Christian S Jensen. 2023. FARGO: Fast maximum inner product search via global multi-probing. *PVLDB* 16, 5 (2023), 1100–1112.
[49] Zhixin Zhou, Shulong Tan, Zhaozhuo Xu, and Ping Li. 2019. Möbius transformation for fast inner product search on graph. In *NeurIPS*. 8218–8229.