Survey & Experimental Analysis of Maximum Inner Product Search Methods

Meghana Nanuvala & Vincent Siddons

1 Introduction

Maximum inner product search (MIPS) plays a crucial role in modern applications like recommendation systems (Feng et al., 2022) and Large-Language Model (LLM) powered search engines (Borgeaud et al., 2021; Lewis et al., 2020) yet efficiently solving this problem in high-dimensional spaces remains a significant challenge. Different tools such as hashing (Charikar, 2002; Shrivastava and Li, 2014; Yan et al., 2018), quantization (Guo et al., 2016; Dai et al., 2020; Guo et al., 2020), graph (Morozov and Babenko, 2018; Zhou et al., 2019; Liu et al., 2019a; Tan et al., 2021), and bandit (Liu et al., 2019b; Yang et al., 2022; Tiwari et al., 2024) methods have been proposed to solve this challenge.

In this survey we describe the current methods in MIPS, compare each method on two commonly used datasets, and finally analyze the results to compare and contrast the strengths of weaknesses of each MIPS approach.

1.1 The Formal Problem

Let $\mathbf{q} \in \mathrm{R}^d$ be a query vector of dimension d, and let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, where $\mathbf{x}_i \in R^d$, denote a database of n vectors. The goal of maximum inner product search (MIPS) is to identify the vector in $\mathcal{X}$ that maximizes the inner product with the query vector $\mathbf{q}$, formally defined as

$$\mathbf{x}^* = \arg\max_{\mathbf{x} \in \mathcal{X}} \langle \mathbf{q}, \mathbf{x} \rangle, \tag{1}$$

where $\langle \mathbf{q}, \mathbf{x} \rangle = \sum_{i=1}^{d} q_i x_i$ denotes the standard Euclidean inner product. In many applications, it is also desirable to retrieve the top-k vectors that yield the largest inner products with $\mathbf{q}$. The top-k formulation is defined as

$$\mathcal{X}_k^* = top{-}k \left\{ \langle \mathbf{q}, \mathbf{x} \rangle : \mathbf{x} \in \mathcal{X} \right\}, \tag{2}$$

where $\mathcal{X}_k^* \subseteq \mathcal{X}$ contains the k vectors with the largest inner products with $\mathbf{q}$.

A *metric* is a function that defines a distance between vectors and satisfies non-negativity, symmetry, and the triangle inequality. The inner product does not induce a proper metric, and consequently, traditional metric-based nearest neighbor search methods (i.e. K-Nearest Neighbors) cannot be directly applied. When all vectors are normalized to unit length, MIPS reduces to maximum cosine similarity search, since

$$\langle \mathbf{q}, \mathbf{x} \rangle = \|\mathbf{q}\|_2 \cdot \|\mathbf{x}\|_2 \cdot \cos\theta = \|\mathbf{q}\|_2 \cos\theta, \tag{3}$$

where θ denotes the angle between $\mathbf{q}$ and $\mathbf{x}$. For large databases and high-dimensional vectors, computing $\langle \mathbf{q}, \mathbf{x}_i \rangle$ for all $\mathbf{x}_i \in \mathcal{X}$ is computationally expensive, motivating the development of approximate or sublinear search methods such as locality-sensitive hashing, graph-based search, and quantization-based approaches.

1.2 A High Level Overview of Each Technique

Hashing These algorithms utilize a hash function for each database vector $\mathbf{x}$, which encodes similarity between database vectors. Algorithms are then applied to these hashed vectors.

Quantization The main idea of these methods is to approximate each database vector $\mathbf{x}$ using a compact code so that inner products with a query vector can be computed efficiently.

Graph-Based Each method in this category reduces the maximum inner product search problem to nearest neighbor search using a similarity graph, where edges connect similar data vectors using some similarity function.

Bandit The main idea of this method is to frame maximum inner product search as a sequential decision problem, where only a subset of candidate vectors or vector coordinates is evaluated to efficiently identify high inner products.

2 Methods

In this section we describe the various methods utilized to solve the MIPS problem by summarizing landmark papers for each technique family.

2.1 Hashing

The problem of searching for a hash function that encodes similarity seems to have been first solved by Charikar (2002), who used a sign random projection algorithm that maps each vector to a compact bit-sketch by thresholding random hyperplane projections, so that the probability of two vectors sharing a bit is proportional to their cosine similarity.

However, the cosine similarity metric normalizes each vector in its calculation so Shrivastava and Li (2014) developed a more efficient and general algorithm based on the inner product, which is a generalization of the cosine similarity as discussed in the earlier section. The algorithm is sublinear and works by applying independent asymmetric transformations, which converts the problem to an approximate nearest neighbor search problem. The authors reported state-of-the art results setting the standard for future work.

Despite this breakthrough, the algorithm for producing these hash functions was still not fully optimized. Yan et al. (2018) developed a new, faster algorithm for designing similarity hash functions by partitioning the dataset into sub-datasets based on the norms of families of data vectors, and then building a hash index for each sub-dataset independently. The only common dataset between (Shrivastava and Li, 2014) and (Yan et al., 2018) was the Netflix dataset, where Yan et al. (2018) were able to match the results of previous baselines while running the algorithm in less time.

2.2 Quantization

All vector quantization methods rely on a certain codebook, such as a cluster centroid, used to store a more compact version of a vector. Guo et al. (2016) extended traditional vector quantization by explicitly optimizing the quantization error for inner products rather than Euclidean distance. Dai et al. (2020) further improved vector quantization for MIPS by explicitly incorporating the norms of database vectors into the quantization process, reducing approximation errors for vectors with widely varying magnitudes. This is done by quantizing the norm separately and jointly optimizing both the norm-weighted distortion and directional error.

Further improving on the above method, Guo et al. (2020) applied anisotropic vector quantization, a method for partitioning vector dimensions based on their variance. The authors combined anisotropic vector quantization (quantizing different dimension bins separately) with asymmetric distance computation (quantizing the database but not the query vector) and optimized multi-stage search (pruning unlikely candidates at each state), which enabled high-speed retrieval with low memory overhead for very large-scale MIPS datasets. The method pioneered by Guo et al. (2020) is the basis for the ScaNN Python library, which implements this efficient technique.

2.3 Bandit

In general, Bandit methods work by framing problems in terms of sequential decisions, where only a subset of candidate solutions need to be evaluated. Liu et al. (2019b) developed a bandit approach to MIPS, which models each database vector as a stochastic "arm" and uses adaptive sampling to prioritize promising candidates, reducing the total number of inner product evaluations. (Tiwari et al., 2024)extended this approach. By sub-sampling coordinates and adaptively evaluating more coordinates for promising database vectors, the authors were able to significantly improve scalability in higher-dimensional datasets. Yang et al. (2022) further demonstrated that embedding approximate MIPS solvers into linear bandit frameworks can achieve sublinear time complexity for arm selection while controlling regret, which is the performance loss from not always selecting the optimal database vector. In empirical settings this leads to a 72x speedup over previous work.

2.4 Graph-Based

A database can be represented by a similarity graph, where similar vectors have edges connecting them with higher weights than less similar vectors. Unlike hashing and quantization methods, vectors do not need to have transformations applied to them. Instead, the MIPS problem is seen as a approximate graph traversal problem. Building on this perspective,Morozov and Babenko (2018) were the first to utilize similarity graphs for the maximal inner product problem, finding that it was a "game changer" in terms of runtime/accuracy. Zhou et al. (2019) improved on the original algorithm by proposing

a "simple but novel" method based on a theorem connecting isomorphisms between subgraphs for inner products.

Hierarchical Navigable Small World (HNSW) graphs, introduced by Malkov and Yashunin (2020), provide an efficient and scalable construction for such similarity graphs by organizing data points into a multi-layer small-world structure that enables logarithmic search complexity in practice. Liu et al. (2019a) made further improvements by avoiding graph walks to irrelevant large norm items by introducing an additional angular proximity graph to initialize the search pool. Finally, Tan et al. (2021) introduced an even more effective algorithm utilizing a norm adjusted proximity graph, which is able to select more meaningful data points for candidate search.

3 Experiment

3.1 Datasets Background

We consider two common data representations drawn from the vision and language domains, which reflect typical workloads in large-scale similarity search and retrieval systems.

SIFT Descriptors: Scale-Invariant Feature Transform (SIFT) descriptors are local image features designed to be robust to scale, rotation, and illumination changes (Lowe, 2004). Due to their high dimensionality and well-defined Euclidean geometry, SIFT features are widely used for evaluating nearest neighbor search algorithms under L2 distance in vision retrieval tasks.

GloVe Word Embeddings: Global Vectors for Word Representation (GloVe) embeddings are dense vector representations that capture semantic relationships between words based on global co-occurrence statistics (Pennington et al., 2014). Their dense structure and semantic similarity properties make them well suited for similarity search based on cosine similarity and Maximum Inner Product Search (MIPS), which are central to modern embedding-based retrieval systems (Lewis et al., 2020; Borgeaud et al., 2021).

3.2 Design

This work studies the design trade-offs of approximate nearest neighbor (ANN) algorithms for large-scale similarity search, with a focus on *Maximum Inner Product Search (MIPS)*. ANN has become a core primitive in modern retrieval systems, including large-scale recommendation and embedding-based information retrieval (Lewis et al., 2020; Borgeaud et al., 2021). We consider three representative ANN families that are widely used in practice:

- **Graph-based methods** (HNSW) (Malkov and Yashunin, 2020; Liu et al., 2019a),
- **Tree and hashing–based methods** (ScaNN) (Guo et al., 2020),
- **Quantization-based methods** (FAISS OPQ+PQ) (Guo et al., 2016).

Each method embodies a distinct design philosophy. HNSW constructs a multi-layer proximity graph to enable fast greedy traversal at query time, trading memory overhead and index construction cost for low query latency (Malkov and Yashunin, 2020). ScaNN combines hierarchical partitioning with asymmetric hashing and selective reordering, leveraging anisotropic vector quantization to balance recall, latency, and memory usage for MIPS workloads (Guo et al., 2020). FAISS OPQ+PQ follows a quantization-based approach inspired by Quantization-based Fast Inner Product Search (QFIPS), compressing vectors into compact codes to significantly reduce memory footprint at the expense of reduced retrieval accuracy (Guo et al., 2016; Dai et al., 2020).

To ensure a fair comparison, all methods are evaluated under the same task definition and similarity metric for each dataset. For MIPS experiments, inner product similarity is used, consistent with prior work on inner product–based retrieval and angular search after normalization(Charikar, 2002; Shrivastava and Li, 2014). Evaluation focuses on the trade-offs between retrieval accuracy, index construction time, query latency, and memory consumption, which are the primary concerns in large-scale ANN systems (Morozov and Babenko, 2018; Tiwari et al., 2024).

3.3 Experimental Settings

3.3.1 Datasets

We evaluate ANN methods on two widely used benchmarks that provide fixed embeddings and exact nearest-neighbor ground truth.

GloVe-100-angular: This dataset consists of approximately 1.18 million 100-dimensional word

embeddings with 10,000 query vectors. It is designed for angular similarity and Maximum Inner Product Search (MIPS), and exact nearest-neighbor ground truth is provided. GloVe-100-angular is a standard benchmark in ANN evaluations and is derived from GloVe word embeddings (Pennington et al., 2014; glo; Guo et al., 2020).

SIFT1M: This dataset contains one million 128-dimensional SIFT descriptors with precomputed ground truth under L2 distance. SIFT1M is a canonical benchmark for Euclidean ANN evaluation and has been extensively used in prior work on nearest neighbor search (Lowe, 2004; tex; Malkov and Yashunin, 2020).

3.3.2 Evaluation Metrics

We report the following evaluation metrics, which are standard in ANN benchmarking and system-level comparisons:

- **Recall@10**, computed against exact nearest-neighbor ground truth under the same similarity metric, following common practice in ANN evaluations (Guo et al., 2016, 2020; Tiwari et al., 2024),
- **Index build time**, measured as total preprocessing and construction time, which reflects offline indexing cost (Malkov and Yashunin, 2020),
- **Query latency**, reported as average per-query time to capture online retrieval efficiency (Guo et al., 2020),
- **Index memory footprint**, measured as the size of the ANN index in memory, which is critical for large-scale deployment (Guo et al., 2016; Dai et al., 2020).

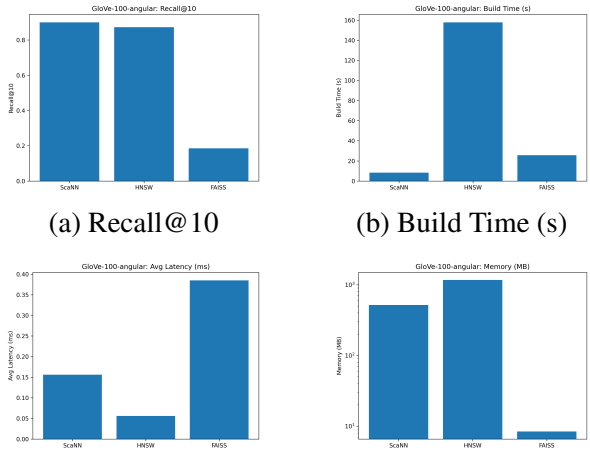
All experiments are conducted on CPU-only environments to ensure consistent and reproducible comparisons across methods, as commonly adopted in prior ANN system studies (Guo et al., 2020; Malkov and Yashunin, 2020).

3.4 Results

3.4.1 GloVe-100-angular (MIPS)

Table 1: Performance comparison on GloVe-100-angular (MIPS).

Method	Recall@10	Build Time (s)	Latency (ms)	Search Time (s)	Memory (MB)
ScaNN	0.9002	8.28	0.156	1.56	517.32
HNSW	0.8729	157.79	0.056	0.56	1168.30
FAISS OPQ+PQ	0.1853	25.67	0.385	3.85	8.41



(a) Recall@10 (b) Build Time (s)

(c) Avg. Latency (ms) (d) Memory (MB, log scale)

Figure 1: Performance metrics on GloVe-100-angular (MIPS).

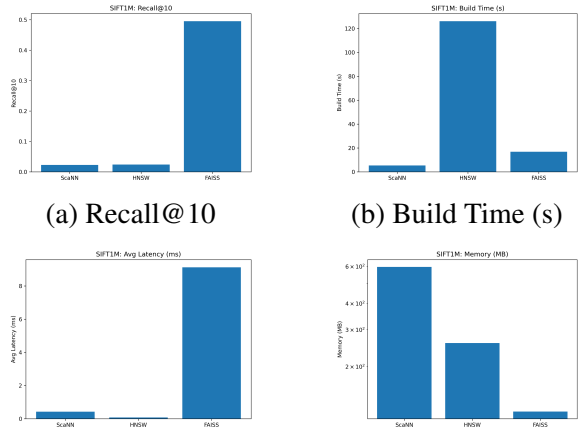
On the GloVe-100-angular benchmark, the evaluated methods exhibit clear trade-offs between accuracy, latency, and memory usage. ScaNN achieves the highest Recall@10 (0.9002), demonstrating strong effectiveness for large-scale MIPS workloads through its combination of partitioning and asymmetric quantization (Guo et al., 2020). HNSW attains slightly lower recall (0.8729) but delivers the lowest query latency, reflecting the efficiency of graph-based greedy traversal at inference time (Malkov and Yashunin, 2020). In contrast, FAISS OPQ+PQ yields substantially lower recall (0.1853) while providing an extremely compact index, consistent with the accuracy–compression trade-offs reported in prior quantization-based approaches (Guo et al., 2016; Dai et al., 2020).

Overall, these results suggest that ScaNN offers the best balance between accuracy and efficiency for MIPS, HNSW is preferable in latency-critical settings with relaxed memory constraints, and FAISS OPQ+PQ is well suited for scenarios where memory efficiency is the primary concern.

3.4.2 SIFT1M

Table 2: Performance comparison on SIFT1M.

Method	Recall@10	Build Time (s)	Latency (ms)	Search Time (s)	Memory (MB)
ScaNN	0.0228	5.36	0.424	4.24	597.47
HNSW	0.0239	126.16	0.077	0.77	259.27
FAISS	0.4954	16.77	9.126	91.26	122.20



(a) Recall@10 (b) Build Time (s)

(c) Avg. Latency (ms) (d) Memory (MB, log scale)

Figure 2: Performance metrics on SIFT1M (L2).

For the SIFT1M benchmark, FAISS attains substantially higher Recall@10 (0.4954) than ScaNN and HNSW under the evaluated settings. This outcome is expected, as SIFT descriptors and their ground truth are defined under Euclidean (L2) distance, which aligns naturally with FAISS's L2-based indexing and quantization mechanisms (Guo et al., 2016; Dai et al., 2020). In contrast, ScaNN and HNSW exhibit markedly lower recall when configured for inner product similarity, reflecting a mismatch between the dataset geometry and the search metric. These results highlight the critical role of metric alignment: ANN methods are most effective when the index construction and query metric are consistent with the intrinsic structure of the data (Morozov and Babenko, 2018; Malkov and Yashunin, 2020).

3.4.3 Overall Observations

Across both datasets, no single approximate nearest neighbor (ANN) method dominates all evaluation metrics. Graph-based approaches favor query-time efficiency through greedy traversal on proximity graphs (Malkov and Yashunin, 2020; Liu et al., 2019a), tree and hashing–based methods balance recall and resource usage via hierarchical partitioning and asymmetric quantization (Guo et al., 2020), and quantization-based methods prioritize memory efficiency through vector compression (Guo et al., 2016; Dai et al., 2020). These findings highlight the necessity of selecting ANN algorithms based on application-specific constraints rather than relying on a universal solution.

Beyond algorithmic trade-offs, our results emphasize the importance of metric alignment in ANN systems. ANN methods perform best when the similarity metric used for indexing and search matches the intrinsic geometry of the dataset, as observed for GloVe-100-angular under Maximum Inner Product Search (MIPS) and SIFT1M under L2 search (Pennington et al., 2014; Lowe, 2004). Metric mismatches can substantially degrade recall, even for highly optimized methods, underscoring the need for application-aware ANN selection that accounts for the interaction between dataset geometry, similarity measure, and indexing strategy (Guo et al., 2016; Liu et al., 2019a).

3.5 Discussion

This study reinforces that approximate nearest neighbor (ANN) indexing is fundamentally a systems-level design problem rather than a universal algorithmic choice. Our results show that different ANN families emphasize distinct trade-offs: graph-based methods such as HNSW prioritize low query latency through efficient graph traversal (Malkov and Yashunin, 2020), tree and hashing-based approaches such as ScaNN achieve strong accuracy–efficiency balance for large-scale MIPS workloads (Guo et al., 2020), and quantization-based methods like FAISS OPQ+PQ favor memory efficiency at the expense of recall (Guo et al., 2016; Dai et al., 2020). Crucially, ANN performance is highly sensitive to metric alignment between data representation and search objective, consistent with prior observations in MIPS literature (Shrivastava and Li, 2014; Morozov and Babenko, 2018). These findings highlight the necessity of jointly considering dataset geometry, similarity metric, and system constraints when deploying ANN systems.

4 Conclusion

This work presented a comparative experimental study of representative ANN methods for Maximum Inner Product Search. Consistent with prior studies (Malkov and Yashunin, 2020; Guo et al., 2020, 2016), our results show that no single ANN technique dominates across accuracy, latency, and memory metrics. Instead, performance depends on metric alignment, dataset geometry, and system constraints. These findings reinforce that ANN indexing is fundamentally a system-level design choice rather than a universal algorithmic solution.

Reproducibility: To support reproducibility, we release all implementation code, dataset preparation scripts, and evaluation pipelines at `https:`

//github.com/meghanaNanuvala/MIPS.

References

Corpus texmex. http://corpus-texmex.irisa.fr/. Accessed: 2025-12-02.

Glove: Global vectors for word representation. https://nlp.stanford.edu/projects/glove/. Accessed: 2025-12-02.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Laurent Sifre, and 1 others. 2021. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*.

Moses S Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 380–388. ACM.

Xiang Dai, Xuanqing Yan, K K W Ng, Jing Liu, and Ji Cheng. 2020. Norm-explicit quantization: Improving vector quantization for maximum inner product search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3845–3853.

Chao Feng, Defu Lian, Xiting Wang, Zheng Liu, Xing Xie, and Enhong Chen. 2022. Reinforcement routing on proximity graph for efficient recommendation. *ACM Transactions on Information Systems (TOIS)*, 40(4):1–25.

Ruoxi Guo, Sanjiv Kumar, Krzysztof Choromanski, and David Simcha. 2016. Quantization-based fast inner product search. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51, pages 482–490.

Ruoxi Guo, Pengcheng Sun, Erik Lindgren, Qi Geng, David Simcha, Fei Chern, and Sanjiv Kumar. 2020. Accelerating large-scale inference with anisotropic vector quantization. In *Proceedings of the International Conference on Machine Learning*, volume 119, pages 3887–3896.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*. NeurIPS 2020 version.

Jing Liu, Xuanqing Yan, Xiang Dai, Yuxin Li, Ji Cheng, and Ming Yang. 2019a. Understanding and improving proximity-graph-based maximum inner product search. *arXiv preprint arXiv:1909.13459*.

Rui Liu, Tian Wu, and Babak Mozafari. 2019b. A bandit approach to maximum inner product search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4383–4390.

David G. Lowe. 2004. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, volume 60, pages 91–110.

Yu. A. Malkov and D. A. Yashunin. 2020. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836.

Stepan Morozov and Artem Babenko. 2018. Non-metric similarity graphs for maximum inner product search. In *Advances in Neural Information Processing Systems*, volume 31.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Anshumali Shrivastava and Ping Li. 2014. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *Advances in Neural Information Processing Systems*, volume 27.

Mengying Tan, Chao Xu, Yufei Zhao, Hongbin Fei, Xue Zhou, and Shiji Li. 2021. Norm-adjusted proximity graph for fast inner product retrieval. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1358–1368.

Mihir Tiwari, Rui Kang, Joon Lee, David Lee, Chris J Piech, Sebastian Thrun, Ilan Shomorony, and Ming Jiang Zhang. 2024. Faster maximum inner product search in high dimensions: Hybrid indexing for mips. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pages 34591–34611.

Xuan Yan, Ji Li, Xiang Dai, Hao Chen, and Ji Cheng. 2018. Norm-ranging lsh for maximum inner product search. In *Advances in Neural Information Processing Systems*, volume 31.

Shiyu Yang, Tianhao Ren, Srinivas Shakkottai, Eric Price, Inderjit S Dhillon, and Sujay Sanghavi. 2022. Linear bandit algorithms with sublinear time complexity. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 25241–25260.

Xue Zhou, Mengying Tan, Chao Xu, and Shiji Li. 2019. Möbius transformation for fast inner product search on graph. In *Advances in Neural Information Processing Systems*, volume 32.