Nearest and Fastest

Free Sample · One Chapter

HNSW versus IVF Indexing for Vector Search

Md Nasim Sheikh

www.nasimstg.dev

About this sample

This is one complete chapter from *Nearest and Fastest*, a short, data-driven field guide to choosing a vector-search index. The numbers here are measured by a reproducible, open-source Docker harness you can run on your own embeddings. The full book explains the IVF and HNSW indexes in detail and turns the comparison into a selection framework. The chapter below, "The Measured Comparison," reports the head-to-head result.

4

The Measured Comparison

Rather than assert which index is faster, this book includes a benchmark that can be run directly. The harness (in the `bench/` directory) builds an HNSW index (`hnswlib`) and an IVF index (FAISS `IndexIVFFlat`) over the *same* vectors, answers the *same* queries against exact ground truth, and records recall, query latency, build time, and index size. Query latency is measured single-threaded for both, so the comparison is like for like.

The run reported here uses 100,000 vectors of dimension 128, drawn as a Gaussian mixture of 1,000 clusters. Uniform-random vectors have no neighbour structure in high dimensions and would misrepresent the problem; clustered vectors mimic the way real embeddings group by meaning. The raw output is in `bench/results.json`, and the appendix gives the exact configuration.

4.1 Both indexes reach near-perfect recall

Table 4.1 reports the headline numbers at a high-recall operating point, together with build time and index size.

Table 4.1. Measured results: 100,000 vectors, $d = 128$, $k = 10$, clustered data (AMD Ryzen 7 5700X, Docker, 2026). Latency is per query, single-threaded. Reproduce with the included harness.

Index	Build (s)	Index (MB)	Latency (ms)	At recall@10
HNSW (hnswlib)	15.7	66.0	**0.050**	0.9997 (ef=100)
IVF (FAISS)	**2.9**	**52.5**	0.064	0.9997 (nprobe=8)

Both indexes reach recall of 0.9997 at $k = 10$. What differs is the price. HNSW reaches that recall at 0.050 ms per query; IVF needs 0.064 ms at its best

high-recall setting, and its latency climbs steeply if `nprobe` is raised further. IVF, in turn, builds more than five times faster (2.9 s versus 15.7 s) and stores a smaller index (52.5 versus 66 MB). Figure 4.1 shows the full trade-off.

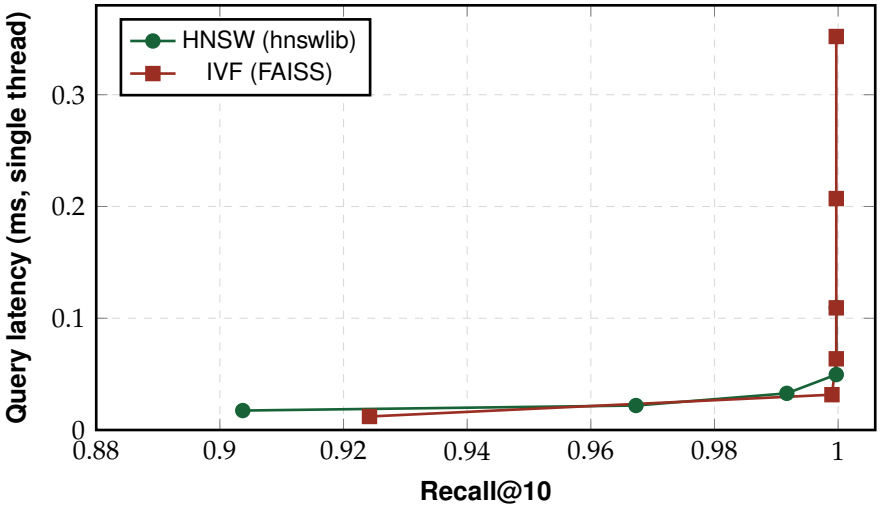

Figure 4.1. Recall versus query latency (lower and further right is better). Each point is an operating setting: `ef` for HNSW, `nprobe` for IVF. Both reach recall ≈1.0, but HNSW holds it at lower latency, while IVF's latency rises sharply as `nprobe` grows past the point where recall has already saturated.

Reading Figure 4.1

Low-recall corner: at recall around 0.92, IVF with a single probe is the fastest point on the chart (0.012 ms), cheaper than HNSW's lowest setting. If a modest recall is acceptable, IVF is very cheap.

High-recall plateau: both reach 0.9997, but HNSW gets there at 0.050 ms while IVF's cheapest 0.9997 point is 0.064 ms, and adding probes only adds latency. HNSW's curve stays flatter as recall approaches one.

Off the chart: HNSW's slower build (15.7 s vs 2.9 s) and larger index (66 vs 52.5 MB) are the price of that query advantage.

4.2 What the numbers mean

The measurement matches the theory. HNSW's logarithmic-hop search keeps query latency low even at the top of the recall range, which is why its curve barely rises approaching recall one. IVF's linear probing means each extra unit of recall near the top costs a whole extra cell scan, which is why its latency fans

upward. The reverse holds off the query path: IVF's single k-means build is far cheaper than constructing a graph, and its lean index costs less memory.

Key Insight

For a fixed high recall target, HNSW tends to answer queries faster; for a fixed build-time or memory budget, IVF tends to win. Neither dominates. The right choice depends on which resource is scarce, which is the subject of the next chapter.

These are measurements on one machine, one dataset, and one scale. They show the *shape* of the trade-off, not a universal ranking. The appendix documents the exact setup, and the harness lets you reproduce the curves on your own embeddings, which is the only comparison that should decide a production choice.

Get the full book

The complete *Nearest and Fastest* covers the IVF and HNSW indexes in full, the selection framework, and a benchmarking protocol, and ships the reproducible Docker harness behind every figure.

www.nasimstg.dev

Open-source benchmark harness included.