Navigating the Vector Search Landscape: ANN Indexes for Modern AI Systems

Ilias Azizi[1][0009−0009−0593−3532], Vassilis Christophides[2][0000−0002−2076−1881], and Themis Palpanas[1][0000−0002−8031−0265]

[1] LIPADE, Université Paris Cité, France
{azizii.ilias@gmail.com, themis@mi.parisdescartes.fr}
[2] ETIS, CNRS, ENSEA, France, IPAL Singapore
vassilis.christophides@ensea.fr

Abstract. Vector databases have become a cornerstone of modern data science and AI applications, enabling the retrieval of relevant data based on semantic meaning and context under strict latency, accuracy, and resource constraints. This demand has strongly shaped the development of approximate nearest neighbor (ANN) methods and accelerated the emergence of new approaches at an unprecedented pace. As a result, vector search is moving beyond static, vector-only indexing techniques toward dynamic, predicate-aware, and workload-aware designs that better reflect the requirements of real deployments. Recent systems increasingly account for updates, query diversity, structured filters, hardware constraints, and resource efficiency, shaping a new generation of vector search solutions for modern data management systems.
This tutorial provides a structured introduction to this evolving landscape from both algorithmic and systems perspectives. It first presents the foundations of vector search and the main ANN indexing families, including space-partitioning, hashing-based, quantization-based, and graph-based methods. It then discusses how these techniques are integrated into modern AI pipelines through hybrid and filtered search, query optimization, adaptive execution, benchmarking methodologies, and index maintenance. Finally, it highlights recent research directions driven by emerging applications, including retrieval-augmented generation, AI agents, memory infrastructure, and energy-aware vector search.

Keywords: Vector Indexing · Approximate Nearest Neighbor Search

1 Introduction

Vector data has become ubiquitous in modern information systems. While vectors have long been used in scientific computing, signal processing, multimedia retrieval, and data analytics, recent advances in machine learning have transformed them into a general-purpose representation for multimodal data [44, 21]. Text documents, images, audio signals, videos, user profiles, products, source code, scientific observations, and even structured records can now be mapped into high-dimensional embedding spaces, where semantic or functional similarity

is reflected by geometric proximity. This shift is mainly driven by the widespread adoption of representation learning and foundation models [7, 58].

Large language models, dense retrievers, vision-language models, and recommender systems increasingly generate embeddings for documents, queries, images, users, and items [34, 35, 49, 17, 58, 51]. Learned representations are also widely used in scientific and industrial applications, including classification, clustering, anomaly detection, data discovery, and data attribution methods in explainable AI [43, 42]. As a result, embedding vectors are now generated at massive scale and on a continuous basis, leading to collections that may contain millions or billions of high-dimensional vectors. Vector search has therefore become a core component of modern data systems, enabling efficient retrieval over these large collections of embeddings [21]. It supports a wide range of applications, including semantic search [35, 34, 58], recommendation [17, 51], multimedia retrieval [49, 58], or data attribution methods in explainable AI [43]. In these applications, data objects are represented as vectors, and queries require retrieving the most similar vectors according to a given similarity function.

Formally, given a dataset $\mathcal{X} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$, a query vector $q \in \mathbb{R}^d$, and a similarity function, vector search aims to retrieve the k nearest neighbors of q in $\mathcal{X}$. A straightforward solution computes the similarity between q and every vector in the dataset and returns the top-k results. However, this exhaustive scan has time complexity $O(nd)$ and becomes impractical at large scale, especially when collections contain millions or billions (n) of high-dimensional (d) vectors [22]. To avoid this exhaustive scan at query time, nearest-neighbor indexes are built to organize the vector collection and accelerate search [19, 47]. In many application settings, strict exactness is less important than meeting latency and throughput requirements; therefore, systems often trade a controlled amount of result accuracy for faster query processing [20, 46].

Approximate Nearest Neighbor (ANN) search addresses this issue by trading a controlled amount of accuracy for substantial gains in latency, throughput, and memory efficiency [6, 50]. This has led to a rich ecosystem of vector indexes. Space-partitioning methods, including tree-based and clustering-based approaches, organize the vector space into regions or subsets so that similar vectors are likely to reside in the same or nearby partitions [9, 19, 47]. Hash-based techniques map vectors into one or multiple hash tables, with the goal that similar vectors collide with high probability and can therefore be retrieved by probing only a small number of buckets [55, 56]. Graph-based methods represent the dataset as a proximity graph, where vertices correspond to data vectors and edges connect vectors that are close according to the target similarity measure; at query time, search traverses the graph greedily or with beam search to progressively move toward the nearest neighbors [5, 39, 4, 52, 25, 41, 14, 40, 24]. Quantization-based approaches mainly reduce the cost of storage and distance computation by encoding vectors into compact representations, enabling faster approximate comparisons and lower memory footprint [30, 33, 1, 26, 27]. These techniques are not mutually exclusive and are often combined in practical systems: for example, quantization can be integrated with inverted files, clustering,

or graph structures to reduce memory usage and accelerate distance computations [53, 1, 52, 48, 4, 60].

Building an efficient vector search system should essentially balance diverse performance metrics such as accuracy, latency, and throughput. This requires careful choices about query execution, including parameter tuning, parallelism, and hardware utilization [1, 37, 48]. Beyond recall and latency, recent work increasingly considers memory footprint, indexing cost, distance computations, hardware utilization, and energy consumption [46, 62, 6]. As vector similarity is frequently combined with structured predicates or metadata constraints, filtered and hybrid vector search has become a core system challenge [36, 2].

Vector search systems must also adapt to evolving workloads and dynamic data. This includes handling insertions, deletions, and updates, as well as improving performance for hard or out-of-distribution queries through workload-aware indexing and index maintenance [13, 29, 59, 31]. At the same time, rigorous benchmarking is needed to evaluate performance across datasets, recall targets, hardware settings, and similarity measures [3, 50, 10]. These issues are becoming even more important with emerging workloads such as RAG, AI agents, long-term memory, and multimodal search, where retrieval quality, freshness, robustness, and energy efficiency directly affect the behavior of downstream AI systems [23, 32, 57, 31, 18].

This tutorial aims to provide a structured introduction to vector indexes from both algorithmic and system perspectives. It first presents the foundations of vector search and the main ANN indexing families, including space-partitioning, and graph-based methods. It then discusses how these techniques are integrated into modern AI pipelines through hybrid and filtered search, query optimization, adaptive execution, and benchmarking methodologies. Finally, it highlights recent research directions for emerging applications such as vector search for LLM-based applications and long-term memory.

Target audience and assumed background. The tutorial targets researchers, graduate students, and practitioners interested in data management, information retrieval, machine learning systems, and vector databases. It is suitable both for newcomers who want a structured introduction to vector indexing and for experienced participants interested in system-level design choices and recent research directions. The assumed background is basic knowledge of algorithms, data structures, and machine learning representations. Familiarity with high-dimensional vectors, similarity measures, or database indexing is useful, but not required. The tutorial introduces the necessary fundamentals before discussing advanced ANN indexing techniques and system-design issues.

Relation to Previous Tutorials. Recent tutorials have addressed faster ANN execution [11], filtered vector search [16], and dense retrieval pipelines [8]. These works focus on specific aspects such as parallelism, streaming, early termination, constrained retrieval, or IR-oriented applications. Our tutorial is complementary: it provides a unified, system-oriented introduction to vector indexes, connecting ANN indexing families with hybrid search, query optimization, benchmarking, energy footprint, and emerging AI-native workloads.

Table 1. Tutorial outline.

Time	Part	Content
10 min	Introduction and Motivation	Role of vector search in modern data systems; applications in semantic search, recommendation, RAG, multimedia, and scientific data; main challenges in scale, latency, accuracy, and energy consumption.
10 min	Problem Definition and Fundamentals	Exact and approximate vector similarity search; distance measures and embedding spaces; quality–efficiency trade-offs in recall, latency, and memory.
30 min	Families of ANN Indexing Techniques	Space-partitioning methods, including trees, hashing, and clustering; subspace-collision methods; graph-based proximity indexes; comparative overview and design trade-offs.
15 min	Hybrid and Filtered Search	Combination of multiple indexing strategies; filtered and constrained vector search; query-aware indexing and query optimization techniques.
10 min	Evaluation and Benchmarking	Measures such as recall, latency, throughput, and cost; benchmark datasets and methodologies; practical evaluation pitfalls.
10 min	Emerging Trends and Research Directions	Vector search for AI agents and large language models; adaptive and learned indexes; AI-native data systems.
5 min	Conclusion and Discussion	Key takeaways, open challenges, and Q&A.

2 Tutorial Scope

In this 1.5-hour tutorial, we cover the foundations, indexing techniques, system-level challenges, and emerging directions of vector search. The tutorial is organized into seven parts, summarized in Table 1.

2.1 Introduction and Motivation

We start by motivating vector search as a building block of modern AI systems. We discuss its role in semantic search [34], recommendation [17], multimedia retrieval [49], Machine Learning [43], and Retrieval-Augmented Generation (RAG) [35, 58]. We formalize the nearest-neighbor search problem and explain why exact search becomes increasingly expensive as vector collections grow to millions or billions of objects. In this context, we introduce approximate nearest-neighbor (ANN) search, where systems trade a controlled loss in accuracy for lower latency, higher throughput, and reduced resource consumption. We highlight the main challenges that make indexing necessary: large-scale collections,

strict latency requirements, accuracy constraints, dynamic workloads, and energy consumption.

2.2 Families of ANN Indexing Techniques

We then review the main families of ANN indexing methods and the principles behind their design. We cover space-partitioning methods, including trees [19, 45], hashing [28], and clustering-based approaches [38, 33], followed by subspace-collision techniques that exploit lower-dimensional projections or partitions of the feature space [28, 56]. We then focus on graph-based indexes [39, 52, 25], where proximity graphs support efficient navigation toward nearest neighbors. This part concludes with a comparative discussion of the design trade-offs across these families, including construction cost, memory footprint, search efficiency, update support, and robustness across datasets.

2.3 Hybrid and Filtered Search

This part discusses vector search in realistic data-system settings, where vector similarity is often combined with structured predicates and metadata filters. We cover filtered and constrained vector search [2, 36], the combination of multiple indexing strategies [4, 63, 54], and query-aware indexing [13, 29]. We also discuss query optimization techniques used to balance selectivity, recall, and execution cost [12, 61]. The goal is to show how ANN search moves from a standalone similarity-search problem to an operator that must interact with predicates, cost models, and system-level constraints.

2.4 Evaluation and Benchmarking

We discuss how vector search systems should be evaluated. We present standard measures, and explain how benchmark datasets and workload choices affect conclusions [3, 6, 50, 10]. Beyond recall and latency, we emphasize the need to report construction time, memory usage, update costs, throughput, tail latency, and energy consumption. We also highlight practical evaluation pitfalls [15], including unfair parameter tuning, ignoring construction and maintenance costs, reporting only average recall, overlooking energy consumption, and using workloads that do not reflect deployment constraints.

2.5 Emerging Trends and Research Directions

We conclude the technical part with emerging research directions. We discuss vector search for AI agents and large language models, where retrieval quality directly affects downstream reasoning and generation. We also cover adaptive and learned indexes, as well as AI-native data systems where vector search becomes a first-class component of the data management stack [23, 32, 57, 31, 18]. Finally, we discuss privacy-preserving and decentralized ANN search, including

settings where vectors, queries, or index ownership are distributed across multiple parties and where retrieval must satisfy privacy, security, or data-sovereignty constraints.

2.6 Conclusion and Discussion

The tutorial ends with a summary of the main takeaways and a discussion of open challenges, including scalability, robustness, dynamic updates, evaluation methodology, privacy, and resource-efficient retrieval.

3 Presenters

Ilias Azizi. is a postdoctoral researcher at Université Paris Cité, LIPADE. His research focuses on scalable similarity search, vector databases, and graph-based approximate nearest neighbor search. He has worked on the design, optimization, and evaluation of efficient vector search methods, with publications in database venues including VLDB and SIGMOD.

Vassilis Christophides. is a Professor of Computer Science at ENSEA, Cergy. He has previously been a Full Professor at the University of Crete, an advanced research scientist at INRIA Paris, and a Distinguished Scientist at Technicolor. His research spans machine learning systems, data science, big data computing, databases, web information systems, and scientific data management. He has authored more than 160 articles in leading journals and conferences, received the 2004 SIGMOD Test of Time Award and several ISWC best paper awards, and has coordinated 22 research projects. He has served in major conference roles, including General Chair of EDBT/ICDT 2014 and Area Chair for ICDE 2016, and is an elected member of the Executive Board of the EDBT Association, as well as a member of ACM and IEEE.

Themis Palpanas. is an ACM Fellow, a Senior Fellow of the French University Institute (IUF), and a Distinguished Professor of Computer Science at Université Paris Cité. He has been the Founding Director of the Data Intelligence Institute of Paris (diiP). He has authored 15 patents, received 3 best paper awards and the IBM SUR award, served as Program Chair for VLDB 2025 and IEEE BigData 2023, General Chair for VLDB 2013, and Editor-in-Chief for BDR. He has been working on high-dimensional vector management and analytics for more than 15 years, has developed several state-of-the-art methods, and has delivered 20 tutorials on these subjects in top international conferences.

Acknowledgments

Supported by EU Horizon projects ARMADA (101168951) and DataGEMS (101188416), and by $Y\Pi AI\Theta A$ & NextGenerationEU project HARSH ($Y\Pi 3TA-0560901$) that is carried out within the framework of the National Recovery and

Resilience Plan "Greece 2.0" with funding from the European Union – NextGenerationEU. This work was granted access to the HPC resources of IDRIS under the allocation 2025-A0191012641 made by GENCI.

References

1. Aguerrebere, C., Bhati, I.S., Hildebrand, M., Tepper, M., Willke, T.: Similarity search in the blink of an eye with compressed indices. Proceedings of the VLDB Endowment **16**(11), 3433–3446 (2023)
2. Ait Aomar, A., Echihabi, K., Arnaboldi, M., Alagiannis, I., Hilloulin, D., Cherkaoui, M.: Rwalks: Random walks as attribute diffusers for filtered vector search. Proceedings of the ACM on Management of Data **3**(3), 1–26 (2025)
3. Aumüller, M., Bernhardsson, E., Faithfull, A.: Ann-benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. In: International Conference on Similarity Search and Applications. pp. 34–49. Springer (2017)
4. Azizi, I., Echihabi, K., Palpanas, T.: Elpis: Graph-based similarity search for scalable data science. Proceedings of the VLDB Endowment **16**(6), 1548–1559 (2023)
5. Azizi, I., Echihabi, K., Palpanas, T.: Graph-based vector search: An experimental evaluation of the state-of-the-art. Proceedings of the ACM on Management of Data **3**(1), 1–31 (2025)
6. Azizi, I., Echihabi, K., Palpanas, T., Christophides, V.: Toward efficient and scalable design of in-memory graph-based vector search. arXiv e-prints pp. arXiv–2509 (2025)
7. Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M.S., Bohg, J., Bosselut, A., Brunskill, E., et al.: On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258 (2021)
8. Bruch, S.: Advances in vector search. In: Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining. pp. 995–997 (2025)
9. Camerra, A., Shieh, J., Palpanas, T., Rakthanmanon, T., Keogh, E.: Beyond One Billion Time Series: Indexing and Mining Very Large Time Series Collections With *i*SAX2+. Knowledge and information systems **39**(1), 123–151 (2014)
10. Ceccarello, M., Levchenko, A., Ileana, I., Palpanas, T.: Evaluating and generating query workloads for high dimensional vector similarity search. In: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining, V.2, KDD. pp. 5299–5310. ACM (2025). https://doi.org/10.1145/3711896.3737383, https://doi.org/10.1145/3711896.3737383
11. Chatzakis, M., Del Gaudio, F., Sideri, S., Palpanas, T.: The quest for faster ann vector search. EDBT (2026)
12. Chatzakis, M., Papakonstantinou, Y., Palpanas, T.: Darth: Declarative recall through early termination for approximate nearest neighbor search. Proceedings of the ACM on Management of Data **3**(4), 1–26 (2025)
13. Chen, M., Zhang, K., He, Z., Jing, Y., Wang, X.S.: Roargraph: A projected bipartite graph for efficient cross-modal approximate nearest neighbor search. arXiv preprint arXiv:2408.08933 (2024)
14. Chen, Q., Wang, H., Li, M., Ren, G., Li, S., Zhu, J., Li, J., Liu, C., Zhang, L., Wang, J.: SPTAG: A library for fast approximate nearest neighbor search (2018), https://github.com/Microsoft/SPTAG

15. Chen, T., Fu, C., Wu, J., Wu, H., Fan, H., Ke, X., Gao, Y., Ni, Y., Zeng, A.: Reveal hidden pitfalls and navigate next generation of vector similarity search from task-centric views:[experiments & analysis]. Proceedings of the ACM on Management of Data **4**(1 (SIGMOD), 1–25 (2026)
16. Chronis, Y., Caminal, H., Papakonstantinou, Y., Özcan, F., Ailamaki, A.: Filtered vector search: State-of-the-art and research opportunities. Proceedings of the VLDB Endowment **18**(12), 5488–5492 (2025)
17. Covington, P., Adams, J., Sargin, E.: Deep neural networks for youtube recommendations. In: Proceedings of the 10th ACM Conference on Recommender Systems. pp. 191–198 (2016). https://doi.org/10.1145/2959100.2959190
18. Deng, Y., You, Z., Xiang, L., Li, Q., Yuan, P., Hong, Z., Zheng, Y., Li, W., Li, R., Liu, H., et al.: Alayadb: The data foundation for efficient and effective long-context llm inference. In: Companion of the 2025 International Conference on Management of Data. pp. 364–377 (2025)
19. Echihabi, K., Fatourou, P., Zoumpatianos, K., Palpanas, T., Benbrahim, H.: Hercules Against Data Series Similarity Search. PVLDB **15**(10) (2022)
20. Echihabi, K., Palpanas, T., Zoumpatianos, K.: New Trends in High-D Vector Similarity Search: AI-driven, Progressive, and Distributed. Proc. VLDB Endow. **14**(12), 3198–3201 (2021), http://www.vldb.org/pvldb/vol14/p3198-echihabi.pdf
21. Echihabi, K., Zoumpatianos, K., Palpanas, T.: Scalable machine learning on high-dimensional vectors: From data series to deep network embeddings. In: WIMS 2020: The 10th International Conference on Web Intelligence, Mining and Semantics. pp. 1–6. ACM (2020)
22. Echihabi, K., Zoumpatianos, K., Palpanas, T., Benbrahim, H.: The Lernaean Hydra of Data Series Similarity Search: An Experimental Evaluation of the State of the Art. PVLDB **12**(2) (2018)
23. Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.S., Li, Q.: A survey on rag meeting llms: Towards retrieval-augmented large language models. In: Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining. pp. 6491–6501 (2024)
24. Fu, C., Wang, C., Cai, D.: High dimensional similarity search with satellite system graph: Efficiency, scalability, and unindexed query compatibility. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)
25. Fu, C., Xiang, C., Wang, C., Cai, D.: Fast approximate nearest neighbor search with the navigating spreading-out graph. Proc. VLDB Endow. **12**(5), 461–474 (2019)
26. Gao, J., Long, C.: Rabitq: Quantizing high-dimensional vectors with a theoretical error bound for approximate nearest neighbor search. Proceedings of the ACM on Management of Data **2**(3), 1–27 (2024)
27. Gou, Y., Gao, J., Xu, Y., Long, C.: Symphonyqg: towards symphonious integration of quantization and graph for approximate nearest neighbor search. Proceedings of the ACM on Management of Data **3**(1), 1–26 (2025)
28. Jafari, O., Maurya, P., Nagarkar, P., Islam, K.M., Crushev, C.: A survey on locality sensitive hashing algorithms and their applications. arXiv preprint arXiv:2102.08942 (2021)
29. Jaiswal, S., Krishnaswamy, R., Garg, A., Simhadri, H.V., Agrawal, S.: Ood-diskann: Efficient and scalable graph anns for out-of-distribution queries. arXiv preprint arXiv:2211.12850 (2022)
30. Jégou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. In: IEEE Transactions on Pattern Analysis and Machine Intelligence. vol. 33, pp. 117–128. IEEE (2011)

31. Jiang, W., Subramanian, S., Graves, C., Alonso, G., Yazdanbakhsh, A., Dadu, V.: Rago: Systematic performance optimization for retrieval-augmented generation serving. In: Proceedings of the 52nd Annual International Symposium on Computer Architecture. pp. 974–989 (2025)
32. Jing, Z., Su, Y., Han, Y.: When large language models meet vector databases: A survey. In: 2025 Conference on Artificial Intelligence x Multimedia (AIxMM). pp. 7–13. IEEE (2025)
33. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with GPUs. IEEE Transactions on Big Data **7**(3), 535–547 (2019)
34. Karpukhin, V., Oguz, B., Min, S., et al.: Dense passage retrieval for open-domain question answering. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (2020), https://aclanthology.org/2020.emnlp-main.550/
35. Lewis, P., Perez, E., Piktus, A., et al.: Retrieval-augmented generation for knowledge-intensive nlp tasks. In: Advances in Neural Information Processing Systems. vol. 33 (2020), https://arxiv.org/abs/2005.11401
36. Lu, D., Caminal, H., Chatzakis, M., Papakonstantinou, Y., Chronis, Y., Jain, V., Özcan, F.: An in-depth study of filter-agnostic vector search on a postgresql database system:[experiments and analysis]. Proceedings of the ACM on Management of Data (SIGMOD) (2026)
37. pgvector maintainers, P.G.D.G..: pgvector: Vector search extension for postgresql with hnsw support. https://github.com/pgvector/pgvector (2023), supports HNSW indexing.
38. Malinen, M.I., Fränti, P.: Balanced k-means for clustering. In: Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshop, S+ SSPR 2014, Joensuu, Finland, August 20-22, 2014. Proceedings. pp. 32–41. Springer (2014)
39. Malkov, Y.A., Yashunin, D.A.: Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. IEEE Trans. Pattern Anal. Mach. Intell. **42**(4), 824–836 (2020)
40. Manohar, M.D., Shen, Z., Blelloch, G., Dhulipala, L., Gu, Y., Simhadri, H.V., Sun, Y.: Parlayann: Scalable and deterministic parallel graph-based approximate nearest neighbor search algorithms. In: Proceedings of the 29th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming. pp. 270–285 (2024)
41. Munoz, J.V., Gonçalves, M.A., Dias, Z., Torres, R.d.S.: Hierarchical clustering-based graphs for large scale approximate nearest neighbor search. Pattern Recognition **96**, 106970 (2019)
42. Myrtakis, N., Castellani, A., Tsamardinos, I.: EDDI: Explainable Data Drift Monitoring using Influence 42nd IEEE International Conference on Data Engineering (ICDE) Montreal (2026)
43. Myrtakis, N., Tsamardinos, I., Christophides, V.: Data glitches discovery using influence-based model explanations. In: Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1. pp. 1068–1079 (2025)
44. Palpanas, T.: Data series management: The road to big sequence analytics. ACM SIGMOD Record **44**(2), 47–52 (2015)
45. Palpanas, T.: Evolution of a Data Series Index: the iSAX Family of Data Series Indexes. Communications in Computer and Information Science (CCIS) **1197** (2020)
46. Pan, J.J., Wang, J., Li, G.: Vector database management techniques and systems. In: Companion of the 2024 International Conference on Management of Data. pp. 597–604 (2024)

47. Peng, B., Fatourou, P., Palpanas, T.: Messi: In-memory data series indexing. In: 2020 IEEE 36th International Conference on Data Engineering (ICDE). pp. 337–348. IEEE (2020)
48. Qdrant Team: Qdrant vector database. https://qdrant.tech/ (2022), accessed: 2025-10-01
49. Radford, A., Kim, J.W., Hallacy, C., et al.: Learning transferable visual models from natural language supervision. In: Proceedings of the 38th International Conference on Machine Learning (2021), https://arxiv.org/abs/2103.00020
50. Simhadri, H.V., Aumuller, M., Douze, M., et al.: Results of the neurips'21 challenge on billion-scale approximate nearest neighbor search. In: Proceedings of Machine Learning Research. vol. 176 (2022), https://arxiv.org/abs/2205.03763
51. Song, L., Pan, P., Zhao, K., Yang, H., Chen, Y., Zhang, Y., Xu, Y., Jin, R.: Large-scale training system for 100-million classification at alibaba. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2909–2930 (2020)
52. Subramanya, S.J., Kadekodi, R., Krishaswamy, R., Simhadri, H.V.: Diskann: Fast accurate billion-point nearest neighbor search on a single node. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. pp. 13766–13776 (2019)
53. Team, E.: Elasticsearch: Distributed, restful search and analytics engine. https://www.elastic.co/elasticsearch (2010), accessed: 2025-10-17
54. Wang, J., Wang, N., Jia, Y., Li, J., Zeng, G., Zha, H., Hua, X.S.: Trinary-projection trees for approximate nearest neighbor search. IEEE transactions on pattern analysis and machine intelligence **36**(2), 388–403 (2013)
55. Wei, J., Lee, X., Liao, Z., Palpanas, T., Peng, B.: Subspace collision: An efficient and accurate framework for high-dimensional approximate nearest neighbor search. Proceedings of the ACM on Management of Data **3**(1), 1–29 (2025)
56. Wei, J., Peng, B., Lee, X., Palpanas, T.: DET-LSH: A locality-sensitive hashing scheme with dynamic encoding tree for approximate nearest neighbor search. Proc. VLDB Endow. **17**(9), 2241–2254 (2024)
57. Wei, J., Xu, Q., Yang, C.: The virtuous cycle: Ai-powered vector search and vector search-augmented ai. arXiv preprint arXiv:2603.09347 (2026)
58. Wu, Z., Chen, X., Pan, Z., Liu, X., Liu, W., Dai, D., Gao, H., Ma, Y., Wu, C., Wang, B., et al.: Deepseek-vl2: Mixture-of-experts vision-language models for advanced multimodal understanding. arXiv preprint arXiv:2412.10302 (2024)
59. Xu, Y., Liang, H., Li, J., Xu, S., Chen, Q., Zhang, Q., Li, C., Yang, Z., Yang, F., Yang, Y., et al.: Spfresh: Incremental in-place update for billion-scale vector search. In: Proceedings of the 29th Symposium on Operating Systems Principles. pp. 545–561 (2023)
60. Yahoo Japan Corporation: Ngt: Neighborhood graph and tree for high-dimensional data. https://github.com/yahoojapan/NGT (2022), accessed: 2024-10-20
61. Zhang, C., J. Miller, R.: Distribution-aware exploration for adaptive hnsw search. Proceedings of the ACM on Management of Data **4**(1 (SIGMOD), 1–27 (2026)
62. Zhang, Y., Liu, S., Wang, J.: Are there fundamental limitations in supporting vector data management in relational databases? a case study of postgresql. In: 2024 IEEE 40th International Conference on Data Engineering (ICDE). pp. 3640–3653. IEEE (2024)
63. Zhao, X., Tian, Y., Huang, K., Zheng, B., Zhou, X.: Towards efficient index construction and approximate nearest neighbor search in high-dimensional spaces. Proceedings of the VLDB Endowment **16**(8), 1979–1991 (2023)