A Comprehensive Survey of Vector Database Management Systems: Foundations, Architectures, and Future Directions

SurveyForge

Abstract— Vector Database Management Systems (VDBMSs) have emerged as a vital tool in efficiently managing high-dimensional data, essential for modern applications in AI and machine learning. This survey articulates the scope and significance of VDBMSs, covering critical dimensions such as indexing methods, hybrid queries, and multimodal data handling. Key findings reveal that advancements in hierarchical navigable small-world graphs and locality-sensitive hashing optimize performance for large-scale applications while managing trade-offs in precision and scalability. Current challenges include real-time index maintenance and energy efficiency in dynamic environments. The survey highlights the integration of VDBMSs with advanced AI systems, such as large language models, to combat issues like hallucinations in generative models, underscoring the potential for retrieval-augmented applications. Looking forward, research must focus on standardizing benchmarks, developing privacy-preserving features, and improving adaptability to diverse data modalities. Collaboration across academia and industry will be pivotal in unlocking the full potential of VDBMSs, positioning them as an integral component in high-dimensional data management and AI-driven innovation.

Index Terms—Vector Database Management, high-dimensional data, retrieval-augmented applications

1 Introduction

Vector Database Management Systems (VDBMSs) represent a crucial evolution in data management, addressing the unique requirements of storing, retrieving, and analyzing high-dimensional data representations known as vectors. Unlike traditional database systems, which excel in managing structured data models, VDBMSs are tailored to handle unstructured and semi-structured data that arise as embeddings in modern machine learning applications. High-dimensional vectors—commonly derived from images, text, audio, and multimodal contexts—serve as numerical abstractions that capture semantic richness, enabling efficient similarity-based querying. This alignment of technology with the burgeoning demands of AI and data-driven decision-making forms the foundation for the development of VDBMSs [1], [2].

The rise of VDBMSs is tightly interwoven with advancements in machine learning and AI, particularly the adoption of embedding techniques like word2vec, GloVe, and Transformer-based models that generate dense vector representations of complex data [3]. These embeddings facilitate tasks such as semantic search, recommendation systems, and natural language processing, defining a new paradigm of querying mechanisms based on vector similarity rather than exact matches. Previous approaches to nearest neighbor searches, while effective algorithmically, fell short in scaling to real-world scenarios where data volumes often reach billions of records [4], [5]. Consequently, VDBMSs emerged as system-level solutions, optimizing key functionalities such as indexing, retrieval, and scalability in high-dimensional spaces [1].

The historical trajectory of VDBMSs captures the shift from academic focus on algorithmic challenges—often centered around indexing structures like k-d trees and locality-sensitive hashing (LSH)—to robust, production-ready systems that integrate advanced graph-based approaches such as Navigable Small World (NSW) and HNSW graphs [6], [7]. A notable development is the integration of learned indexes, which leverage machine learning to improve search accuracy and reduce memory overhead [8]. Furthermore, advancements in hardware optimizations, including GPU and FPGA acceleration, address the computational demands of large-scale vector queries, achieving high throughput and low latency [9], [10].

Despite their growth, VDBMSs face significant challenges. The curse of dimensionality makes traditional measures of similarity, such as Euclidean distance, increasingly unreliable as vector dimensionality rises [11]. Balancing the trade-offs of accuracy, resource utilization, and latency in approximate nearest neighbor search (ANNS) remains a central focus, with recent innovations in vector quantization and hybrid architectures yielding promising improvements [12], [13]. Additionally, the integration of VDBMSs into hybrid query models that support both structured and vectorized data poses an ongoing difficulty, especially when scalability and real-time requirements converge [14].

Emerging trends highlight the convergence of VDBMSs with large-scale neural architectures, such as Large Language Models (LLMs), whereby vector databases serve as embedding repositories for Retrieval-Augmented Generation (RAG) workflows [15], [16]. This fusion underscores the evolving importance of VDBMSs as foundational infrastructure for modern AI ecosystems. To unlock their full potential, future research must focus on innovative indexing designs, adaptive query optimization strategies, and a unified benchmarking framework tailored to the diverse requirements of vector-based workloads [1], [17].

In conclusion, VDBMSs bridge fundamental gaps in high-dimensional data management, transforming how information is stored, retrieved, and leveraged in contemporary workflows. By synthesizing advances in algorithms, systems, and hardware, they continue to define a critical pillar of modern computational frameworks, driving innovation across industries and research domains [1], [18].

2 Core Architecture and Data Management in Vector Database Systems

2.1 Data Representation and Modeling

The representation and modeling of high-dimensional data as vectors lie at the core of vector databases, enabling their ability to perform efficient similarity-based querying. This subsection examines the methodologies used to represent data as vectors, strategies to preserve their semantic relationships, and techniques to model these representations efficiently within vector databases.

High-dimensional vectors are mathematical representations of complex data, such as text, images, or audio, embedding them into a continuous space that captures their semantic or contextual similarities. Typically, these vectors are dense and fixed-length, with each dimension encapsulating a specific feature of the data. Sparse vector representations, though less commonly used, might be practical in specific domains where data characteristics inherently contain numerous zero elements [19]. Neural embedding models such as BERT, word2vec, and ResNet are often employed to transform raw data into these vector representations. For example, word2vec encodes textual data into dense vectors that maintain proximity in the vector space for semantically similar words [3]. Domain-specific embeddings, such as graph-based methods for knowledge graphs, allow the representation of structured data while encoding its relational characteristics [18].

Dimensionality reduction techniques play a pivotal role in vector modeling to mitigate the "curse of dimensionality" and enhance both storage and query performance. Principal Component Analysis (PCA) simplifies vectors by projecting them into a lower-dimensional space while preserving variance. Autoencoders achieve a similar outcome through nonlinear transformations by training neural models to compress and reconstruct data efficiently [20]. Quantization techniques, such as product quantization (PQ) and Locally-Adaptive Vector Quantization (LVQ), further optimize storage by encoding vectors into compact representations with minimal degradation to similarity precision [21]. These methods are particularly advantageous for billion-scale datasets, offering trade-offs between memory efficiency and query accuracy.

As modern applications often deal with multimodal data—such as image-text or audio-text pairs—adaptations must be made to accommodate cross-modal retrieval. For instance, unified embedding spaces enable multiple modalities to share a coherent latent space, facilitating cross-modal similarity search [15]. However, challenges remain in achieving robust alignment between modalities while maintaining query efficiency.

A critical trade-off in data modeling arises between the fidelity of similarity measurement and computational efficiency. Exact similarity measurements (e.g., cosine similarity, inner product) demand significant computational resources for high-dimensional vectors, particularly at scale. Consequently, vector databases commonly employ approximate nearest neighbor (ANN) methods using tree-based, graph-based, or learned index structures [4], [22]. These approaches leverage probabilistic approximations to prioritize efficiency, though often at the cost of minor reductions in result accuracy.

Emerging trends suggest integrating learned representations with hybrid optimization techniques that blend classical approximation strategies with adaptive learning methods. For example, models that dynamically refine embeddings or adapt dimensionality during query execution are gaining attention for their context-aware flexibility. Future research must also address challenges such as balancing multimodal data alignments and achieving energy-efficient vector operations in increasingly large-scale deployments [12], [23].

In conclusion, the representation and modeling of vectorized data are foundational to the operational efficiency of vector databases. Continuous advancements in embedding techniques, vector compression, and dimensionality reduction underscore the field's dynamism, while emerging challenges offer fertile ground for future innovation.

2.2 Indexing and Search Optimization

The optimization of indexing and search mechanisms forms a cornerstone for enabling efficient similarity queries in vector databases, especially given the dual imperatives of handling high-dimensional data and managing resource constraints. Building on the foundational vector modeling techniques discussed previously, this subsection delves into the predominant indexing techniques—graph-based, hashing-based, and tree-based strategies—while examining the trade-offs these approaches pose in terms of accuracy, speed, and resource utilization. Additionally, it highlights the evolving innovations in dynamic index maintenance, laying the groundwork for seamless integration with storage optimization and compression methodologies discussed in subsequent sections.

Graph-based indexing methods, exemplified by Hierarchical Navigable Small-World (HNSW) graphs, represent a state-of-the-art approach for approximate nearest neighbor (ANN) search. HNSW achieves efficient search operations by constructing hierarchical layers of proximity graphs that facilitate rapid traversals and ensure high recall while mitigating computational complexity [24]. Despite their scalability for massive datasets, these methods are memory-intensive due to dense graph structures and require sophisticated algorithmic fine-tuning to balance indexing overhead with query latency. Hybrid adaptations that combine neighborhood graphs with auxiliary structures, such as bridge graphs, have demonstrated enhanced accuracy and scalability, particularly for billion-scale data scenarios [24].

Hashing-based techniques, particularly Locality-Sensitive Hashing (LSH), address the dimensionality challenge by projecting high-dimensional vectors into a lower-dimensional hash space. This reduction enables sublinear query times by organizing similar vectors into

shared hash buckets. More refined variants, such as multi-probe LSH and asymmetric feature maps, further optimize recall and precision trade-offs while managing resource demands [25]. These techniques are particularly well-suited for static datasets; however, their ability to adapt to dynamic data updates lags behind, making them less ideal for applications that demand frequent indexing adjustments.

Tree-based structures like KD-Trees and R-Trees employ hierarchical partitioning schemes to organize data, offering efficient search paths for moderately high-dimensional datasets. While these approaches excel in lower-dimensional scenarios, their performance suffers under high dimensionality due to the "curse of dimensionality." Recent innovations combining tree-based structures with vector quantization or learned models have mitigated this limitation, unlocking new opportunities for handling more complex, multidimensional datasets [8], [26].

Dynamic index maintenance is critical for vector databases that must accommodate frequent data updates and real-time query processing. Techniques such as degree-adjusted graph optimization, incremental indexing, and adaptive partition reorganization have emerged as viable solutions for maintaining indexing efficiency without significant computational overhead [6]. For instance, locality-aware partitioning strategies streamline data movement during updates, minimizing latency and enhancing real-time indexing performance [27].

Emerging trends in indexing innovations are increasingly leveraging machine learning. Learned indices, for example, employ predictive models to forecast data organization and traversal paths, offering promising gains in accuracy and efficiency [8], [28]. However, the applicability of such approaches hinges on their ability to generalize across diverse datasets, presenting an ongoing challenge that necessitates further exploration.

In conclusion, optimizing indexing mechanisms requires a careful balance of system-level trade-offs and application-driven priorities. Hybrid frameworks that blend graph-based precision, hashing efficiency, and tree-based scalability are emerging as essential tools for addressing the diverse needs of modern vector database deployments. Future efforts should focus on enhancing index adaptability to dynamic updates while maintaining real-time performance, ensuring seamless compatibility with compression-optimized storage solutions. Furthermore, incorporating hardware-accelerated indexing techniques holds immense potential for boosting overall efficiency, aligning with the increasingly resource-intensive demands of large-scale, high-dimensional similarity search tasks.

2.3 Vector Compression and Storage Solutions

Efficient management of storage resources is central to vector database systems, given the sheer scale and dimensionality of vector embeddings. Vector compression and storage solutions aim to reduce memory footprints and optimize storage without sacrificing retrieval accuracy or the performance of high-dimensional similarity search tasks. This subsection addresses key strategies, including quantization, efficient storage formats, hybrid compression models, and emerging hardware-aware methodologies.

Vector quantization is a cornerstone of compression techniques, allowing high-dimensional vectors to be stored or approximated using compact representations. Methods such as scalar quantization map vector components to discrete levels, minimizing storage needs, while advanced techniques like product quantization (PQ) partition vectors into subspaces and encode each subspace independently. PQ enhances efficiency by facilitating approximate nearest neighbor (ANN) searches in compressed domains and has demonstrated its efficacy on billion-scale datasets for practical indexing systems [5], [13]. Binary quantization, on the other hand, transforms vectors into binary codes, significantly reducing storage at the cost of limited precision, making it optimal for applications with strict memory constraints [12], [21]. These methods illustrate the trade-offs between compression ratio and retrieval fidelity, where higher compression often causes minor compromises in accuracy.

Memory-efficient storage formats, tailored to high-dimensional data, combine lossy and lossless principles to balance resource utilization with retrieval integrity. Locally-Adaptive Vector Quantization (LVQ), for instance, achieves competitive efficiency through per-vector scaling and scalar quantization, reducing effective bandwidth needs without noticeable accuracy degradation. Such advancements are pivotal in systems needing low latency or extensive deployments across edge devices [12]. Systems like SPANN further exemplify efficient memory-disk hybridization using a hierarchical structure to store smaller centroid data in memory and larger, compressed posting lists on disk. This design ensures scalability and provides high recall with minimal latency [13].

Hybrid compression approaches adapt dynamically to varying query demands and mixed workloads. These models integrate quantization techniques with collaborative filtering or metadata fusion, ensuring that storage architectures adjust to evolving datasets while supporting multimodal searches [29]. Moreover, for data with inherent dynamicity, methods such as dynamic indexing combined with streaming-optimized quantization improve storage efficiency while sustaining high throughput under fluctuating query loads [12].

Hardware-aware compression techniques represent a growing frontier. Compression frameworks designed to leverage GPUs and SSDs for on-the-fly vector decompression and query acceleration enhance performance, especially in graph-based structures like NSG and SSG [7]. Emerging paradigms such as persistent memory-based systems (e.g., APEX) integrate machine-learned indices with PM-optimized layouts, achieving orders-of-magnitude improvements in throughput while substantially reducing memory footprint [30]. These methods signify how modern hardware solutions are reshaping storage efficiency.

Challenges remain, including managing the trade-offs between compression-induced errors and system performance under hybrid query workloads that combine vector and relational data. Future research should prioritize adaptive, workload-specific compression strategies and quantization methods tailored for federated or decentralized systems. At the convergence of machine learning and storage systems, learnable compression models remain an open av-

enue to intelligently parameterize quantization granularity based on data distribution, paving the way for precision-scalable storage solutions in the next generation of vector databases.

2.4 Scalability and Distributed Architectures

Scalability in vector database systems is a pivotal characteristic, particularly as applications increasingly demand the ability to store, query, and index massive high-dimensional datasets. To meet these demands, modern vector databases employ distributed architectures capable of efficiently handling exponential data growth and distributed workload complexities, all while ensuring low-latency response times. This subsection explores the key strategies, trade-offs, and emerging trends in achieving scalability and distributed functionality in vector databases, forming an essential foundation for large-scale high-dimensional similarity search.

Central to distributed vector database systems is the concept of sharding, where data is partitioned across nodes in a cluster. Geometric partitioning methods, such as space-partitioning trees (e.g., KD-Trees) and proximity graphs like navigable small-world networks, are commonly used to minimize the computational cost of cross-node queries. Advanced techniques like the bounded quadrant system [31] integrate geometric optimizations with spatial compression, further enhancing scalability without significantly compromising retrieval performance. In scenarios involving skewed or evolving data distributions, dynamic partitioning approaches adaptively redistribute data across shards, maintaining system responsiveness at substantial computational overheads during rebalancing [32].

Efficient query routing and load balancing are equally critical for maintaining performance in distributed environments. Systems such as Starling [27] demonstrate the potential of locality-aware graph layouts to reduce disk I/O during query execution, achieving high throughput with minimal precision degradation. Similarly, workload-aware architectures like Multi-IndeXing-RS [33] align query patterns with optimal index placement, ensuring reduced latency for high-demand operations. A significant challenge lies in managing hybrid query workloads—those blending similarity-based and relational predicates—which require advanced query planners to optimize distributed execution workflows while leveraging both primary and secondary indices [14].

Consistency models are another pivotal dimension in ensuring system scalability. Relaxed consistency guarantees, such as eventual or delta consistency, are often adopted by systems like Manu [10] to minimize synchronization overheads among nodes, enabling elastic scaling at the expense of immediate consistency. However, applications such as financial fraud detection or real-time decision systems, which demand stricter guarantees, continue to push for innovations balancing these trade-offs.

Elastic scaling has become an increasingly important feature, particularly with the rise of cloud-native database ecosystems. Infrastructure-agnostic frameworks, such as Adaptive Multilinear Meshes [34], enable vector databases to dynamically scale across nodes by leveraging lightweight indexing and compression techniques to reduce the memory footprint. In parallel, emerging second-tier memory technologies like RDMA and CXL are bridging the gap between fine-grained query patterns and the coarse data access patterns inherent to SSD-based storage [35].

Fault tolerance mechanisms are indispensable for distributed systems managing terabyte- to petabyte-scale vector datasets. To ensure robustness, many systems integrate replication with coded redundancy techniques, such as clique-repair strategies [36], to minimize recovery time from node failures. Distributed repair frameworks employing Reed-Solomon codes have demonstrated bandwidth savings of up to 20

Looking ahead, emerging research is likely to delve deeper into interoperability between distributed systems and federated machine learning pipelines, facilitating cross-site vector data management and computation without reliance on centralized storage. The integration of learned data partitioning methods and novel paradigms like quantum vector databases holds significant potential to revolutionize existing scalability frameworks, particularly for handling dynamic, high-dimensional multimodal workloads [15]. Ultimately, balancing trade-offs between scalability, query latency, resource efficiency, and consistency remains a key area of innovation, as the scope and impact of vector databases continue to expand in the era of machine learning-driven applications.

2.5 Hardware Acceleration and Optimization

The integration of modern hardware accelerators has become a cornerstone for optimizing the performance of vector databases in executing high-dimensional similarity search at scale. By leveraging the parallelism and specialized computational capabilities of accelerators like GPUs, TPUs, FPGAs, and emerging ASIC-based architectures, current systems achieve unprecedented throughput, reduced query latencies, and scalability for billion-scale vector datasets. This subsection delves into these optimizations, analyzing diverse hardware strategies, future trends, and their trade-offs.

GPUs are the most pervasive acceleration technology in vector databases due to their immense parallel processing power and support for high-bandwidth memory hierarchies. GPUs significantly expedite computationally expensive operations such as L_2 norm calculations, Hamming distance estimations, and inner product computations for Approximate Nearest Neighbor (ANN) search. For instance, GPU-accelerated libraries such as Faiss [5] demonstrate up to 6.7× speed improvements over traditional CPU-based implementations while seamlessly handling billion-scale datasets. However, limitations such as energy consumption and memory capacity constraints necessitate hybrid architectures combining GPUs with complementary optimizations like hierarchical memory management or disk-based offloading [13].

FPGAs and ASICs, on the other hand, offer tailored, energy-efficient solutions by implementing specialized circuits to optimize vector workloads such as encoding, decoding, and similarity computations. FPGAs leverage configurable logic blocks to pipeline-intensive computational graph traversal tasks effectively, offering acceleration at significantly lower power costs compared to GPUs. For

example, experiments have outlined their utility in reducing latency for dynamic structural similarity queries, particularly in modular systems [37]. ASICs, although more rigid in design, excel in throughput for fixed ANN workloads where predictable query patterns dominate, with systems like Hamming-oriented ASIC-based processors achieving orders-of-magnitude energy savings [37].

In addition to compute acceleration, modern vector databases exploit hierarchical memory architectures, such as high-bandwidth memory (HBM), caching, and multi-level storage. Systems like SPANN [13] combine in-memory centroids with disk-based posting list structures, employing query-aware pruning and cluster augmentation techniques to balance memory and SSD usage effectively. Innovations like Locally-Adaptive Vector Quantization (LVQ) [21] further enhance memory efficiency while maintaining high search precision.

Emerging trends in hardware acceleration include the confluence of neuromorphic computing architectures, particularly for sparse high-dimensional embeddings, and the potential integration of quantum-vector hybrid processing for ANN computations [38]. Moreover, future advancements may focus on heterogeneous distributed systems where multiple accelerators (e.g., GPU-FPGA hybrids) collaborate seamlessly to optimize performance under diverse workload distributions.

While hardware advances enable extraordinary computational gains, challenges persist. These include the complexity of tuning heterogeneous systems [39], the energy implications of accelerator-intensive setups [40], and ensuring compatibility with emerging standards. Thus, balancing cost, power consumption, and query precision remains a focal point for future research.

Looking forward, the inclusion of adaptive runtime tuning frameworks and the optimization of hardware-aware vector quantization algorithms will further enhance the applicability of accelerators for vector databases. Coupled with the synergistic evolution of vector-specific benchmarks [1], these advancements are poised to redefine the boundaries of real-time, large-scale high-dimensional search.

2.6 Adaptive Maintenance and System Evolution

Efficiently adapting vector database systems (VDBs) to dynamic data distributions, evolving query patterns, and changing user demands is fundamental to ensuring their sustained performance and utility in modern applications. The ability to maintain adaptive and responsive systems plays a pivotal role in minimizing operational disruptions while keeping pace with the demands of ever-changing workloads. In this section, we delve into the core techniques, trade-offs, and emerging trends in adaptive maintenance and system evolution for VDBs.

A primary aspect of adaptivity is the dynamic reconfiguration of indexing structures to align with shifting workloads. Advanced techniques such as dynamic index optimization and hybrid partitioning are designed to accommodate the heterogeneity and temporal evolution of input data. For example, navigable proximity graphs [41] dynamically adjust their graph structures to efficiently prioritize frequently accessed data points, striking a balance between precision and performance. Similarly, systems leveraging learned index structures, such as LIMS [26], recalibrate embedding partitions to reflect changes in data clustering or distribution, reducing indexing overhead without compromising result accuracy.

Another critical aspect lies in the task of updating vector embeddings, especially critical when embedding representations are derived from evolving machine learning models. Incremental update methodologies, including delta encoding and multiversion concurrency control, facilitate efficient embedding updates, as demonstrated in systems like Manu [10]. In real-time applications, strategies such as locally-adaptive vector quantization [12] efficiently integrate new data streams without necessitating resource-intensive recomputation, thereby preserving system responsiveness and computational efficiency.

The proactive monitoring and detection of system anomalies are equally crucial for maintaining adaptive performance. Lightweight probabilistic structures, such as those described in [42], provide scalable solutions for identifying performance deviations caused by adversarial inputs, distribution shifts, or hardware malfunctions. These tools enable early detection at a low computational cost, helping to ensure continuous system reliability.

Looking ahead, the evolution of VDBs to accommodate emerging workload demands introduces additional complexities, particularly in the context of federated systems and decentralized environments. These paradigms demand consistent synchronization and effective collaboration mechanisms. Research on distributed systems [43] indicates a growing focus on federated learning and privacy-preserving protocols as means to manage global-scale operations. Moreover, future advances in quantum computing and neuromorphic architectures may redefine foundational adaptive paradigms by unleashing new algorithmic capabilities tailored to handle high-dimensional vector workloads [38].

In summary, adaptive techniques in vector database systems offer promising solutions to evolving data and query landscapes, but they invariably involve trade-offs between computational cost, latency, and system complexity. Tools like VDTuner [39], which leverage machine learning for automated parameter optimization, can provide critical support for maintaining efficiency across diverse workloads. As the field progresses, addressing key challenges in scalability, real-time embedding updates, and the integration of novel hardware architectures will define the next frontier of innovation, driving adaptive VDB systems toward ever-greater robustness and efficiency.

3 Query Execution, Optimization, and Hybrid Search Paradigms

3.1 Similarity Scoring and Distance Computation Paradigms

Similarity scoring and distance computation paradigms represent the foundational methods enabling effective vector similarity search in high-dimensional spaces. They serve as the computational core of query execution, transforming abstract notions of "closeness" into measurable metrics for

diverse applications such as information retrieval, recommendation systems, and multimodal search. This subsection delves into the traditional approaches, emerging methodologies, and their applicability to various datasets, detailing trade-offs and assessing their impact on vector database performance.

Traditional distance metrics like Euclidean distance and cosine similarity remain widely employed in vector databases due to their interpretability and computational simplicity. Euclidean distance, defined as $||\mathbf{q} - \mathbf{v}||_2$, measures the geometric distance between vectors $\mathbf{q}$ (query) and $\mathbf{v}$ (database item) and excels in applications requiring absolute spatial precision, such as image retrieval or clustering [22]. Cosine similarity, $1 - \frac{\mathbf{q} \cdot \mathbf{v}}{||\mathbf{q}||_2 ||\mathbf{v}||_2}$, is dominant in text and multimodal embeddings, capturing angular relationships that reflect semantic similarity rather than magnitude. Despite their strengths, traditional metrics can grow computationally prohibitive with increasing vector dimensionality or dataset size, necessitating scalable accelerations through approximation or compression techniques [4].

Emerging techniques augment these paradigms by incorporating data-driven or domain-specific properties. Learned distance functions, which leverage neural networks to model similarity, provide flexible and adaptive scoring tailored to specific application domains, such as medical imaging or personalized recommendations, where embeddings may not naturally adhere to traditional Euclidean space [8], [26]. These approaches, while more accurate in certain scenarios, require significant training overhead and can pose challenges for generalization across diverse datasets.

Another advancing trend is the hybridization of scoring approaches to balance strengths. For instance, maximum inner-product search (MIPS), $max(\mathbf{q} \cdot \mathbf{v})$, efficiently identifies dominant features shared between vectors and pairs well with approximate nearest neighbor algorithms for applications requiring soft relevance ranking, such as e-commerce and content delivery [7], [11]. Additionally, combinations of similarity and dissimilarity measures, like those proposed in DisC diversity scoring, improve result diversification by balancing tight local clusters with global representation [44].

Scalability remains a significant challenge in designing efficient distance computation paradigms. Lightweight approximations through quantization or hashing techniques have demonstrated success in reducing computational costs in billion-scale datasets without significantly compromising precision [12], [21]. These strategies exploit the underlying statistical structure of high-dimensional spaces, such as sparsity or clustering, to achieve scalability. However, they must contend with potential trade-offs in recall accuracy, particularly in safety-critical domains.

Looking forward, establishing universal scoring models that can seamlessly adapt to multimodal data while maintaining computational tractability represents an open research direction. Hybrid models that unify traditional metrics, learned representations, and domain-specific adaptations hold promise for driving innovation in this space [1], [18]. Furthermore, exploring hardware co-design, as with FPGA-accelerated similarity searches, could unlock further performance gains for real-time, high-throughput applications [9]. Ultimately, the continuous refinement of similarity scoring paradigms is key to expanding the usability and scalability of vector databases across diverse and dynamic operational demands.

3.2 Query Optimization Techniques for Vector Retrieval

The optimization of query execution in vector retrieval systems is essential for delivering scalable, low-latency performance while balancing resource efficiency and accuracy. Within the context of high-dimensional data, query optimization techniques rely on approximation, pruning, caching, and emerging machine-learning-driven strategies to enhance system effectiveness without significantly compromising result quality.

Approximate Nearest Neighbor Search (ANNS) techniques are central to vector query optimization, reducing the need for exhaustive similarity calculations and enabling faster lookups. Methods such as Locality-Sensitive Hashing (LSH) partition vector spaces probabilistically to achieve sub-linear query times, albeit with some precision loss [6], [45]. Graph-based techniques, notably Hierarchical Navigable Small-World (HNSW) networks, organize vectors into proximity graphs that facilitate highly selective nearest-neighbor searches, demonstrating an effective balance between retrieval speed and accuracy [5], [24]. These techniques, while crucial for speed-up, highlight important trade-offs, particularly in cases where slight sacrifices in accuracy are tolerable.

Pruning strategies further enhance efficiency by selectively limiting the scope of similarity calculations. Techniques such as early stopping and relevance-based pruning halt computations once intermediate results sufficiently approximate a query's top-k neighbors, thereby reducing redundant operations and decreasing latency [17]. Dynamic query pruning refines this approach by tailoring thresholds based on query features or intermediate results, yielding significant runtime improvements without degrading precision [46]. These strategies are especially impactful in large-scale deployments managing vast vector collections.

Caching provides another layer of optimization, leveraging patterns in query workloads to store frequently accessed results or intermediary computations for reuse. Semantic caching, for instance, reduces redundant operations by exploiting recurring query characteristics [10]. For time-critical applications, caching vector embeddings and their precomputed nearest neighbors significantly accelerates response times, benefiting scenarios like recommendation systems or retrieval-augmented models [12].

Recent advancements incorporate machine learning to dynamically enhance query execution. For example, learned query adaptation frameworks adjust vector representations or reconfigure index parameters based on observed query patterns, unlocking notable efficiency improvements [8], [26]. Similarly, user feedback-driven models refine retrieval precision over time by embedding personalized or context-aware weights into the optimization process [47]. These techniques introduce a feedback loop conducive to ongoing refinement and adaptation without manual intervention.

Despite progress, challenges persist in optimizing vector queries at scale. High-dimensional sparsity exacerbates

computational inefficiencies, amplifying the "curse of dimensionality" and diminishing the efficacy of traditional approximation and pruning strategies in certain domains [22]. Additionally, ensuring robustness amidst dynamic dataset updates and distributional shifts in query embeddings remains an open problem [12]. Progress in hybrid optimization frameworks combining model-driven techniques with hardware-accelerated solutions, such as FPGA-optimized similarity computations, holds promise for addressing these limitations [27], [46].

In summary, query optimization in vector retrieval systems integrates approximation, pruning, caching, and learning-based strategies to achieve an increasingly sophisticated trade-off between computational efficiency and precision. These innovations align with the broader challenges of similarity scoring, discussed earlier, while seamlessly laying the groundwork for hybrid query processing frameworks detailed in subsequent sections. Continued research in this area is crucial for addressing scalability bottlenecks and evolving operational demands across diverse and data-intensive workflows.

3.3 Hybrid Query Processing for Structured and Unstructured Data

Hybrid query processing in vector database management systems aims to seamlessly integrate vector-based similarity retrieval with traditional structured query mechanisms, such as SQL, to support applications requiring both semantic similarity and relational data constraints. This capability is pivotal in scenarios where unstructured data (e.g., text, images, embeddings) and structured attributes (e.g., metadata, categorical fields) coexist, such as personalized recommendations, fraud detection, and knowledge graph augmentation, as highlighted in [14].

One foundational approach for hybrid query processing involves the synthesis of vector indices, such as hierarchical navigable small-world graphs (HNSW), with relational indexing techniques like B+-trees or learned indexes. Native hybrid query frameworks, such as those illustrated in [41], exploit composite indices that jointly structure the structured and unstructured query parameters. Through joint pruning mechanisms, these frameworks not only accelerate query execution but enhance result accuracy by aligning vector similarity computations with relational attribute filtering. Nevertheless, the computational overhead of maintaining integrated indices across dynamic workloads and diverse data distributions remains a performance bottleneck.

Another strategy is to separate vector retrieval and attribute filtering as independent phases, merging results at the final stage. While conceptually simpler, this method often sacrifices efficiency and increases latency due to redundant retrieval operations across independent query spaces [1]. Attempts to enhance this method include optimization heuristics for query routing and partitioning vector search spaces based on predicate conditions [48]. Despite these strides, balancing computational parity between the two phases under hybrid workloads remains challenging.

The fusion of relational and semantic retrieval also depends on adaptive index designs. Techniques like learned multi-dimensional indexes [28] use machine learning models to predict access patterns and optimize both vector similarity and structured filtering. Such systems have been shown to reduce search latency while scaling efficiently under fluctuating workloads, though they often lack robustness for complex hybrid constraints.

At the frontier, methods that dynamically augment hybrid queries with semantic context represent significant innovations. Embedding metadata as auxiliary features enhances the interpretability of unstructured query results, as demonstrated in [49]. This approach, however, requires substantial integration of domain-specific knowledge and robust metadata management.

Emerging challenges include establishing optimal trade-offs between result accuracy and execution cost. Continuous updates to vector embeddings and relational data necessitate real-time index reconfiguration techniques, which have been partially addressed through dynamic systems such as [50]. Additionally, the lack of standardized evaluation benchmarks tailored for hybrid queries complicates comparative analyses across architectures.

Future research must address the scalability of hybrid query engines in distributed and federated environments, explore cross-modal retrieval techniques that unify disparate data modalities (e.g., images and relational attributes), and advocate for interoperability standards. Moreover, interdisciplinary advancements, primarily within learned systems and hardware acceleration, can bridge the computational gap, ensuring hybrid search paradigms remain efficient at scale.

3.4 Real-Time and Low-Latency Query Resolution

Real-time and low-latency query resolution is pivotal for effectively managing high-dimensional vector data in dynamic environments, especially as applications increasingly demand seamless updates, high throughput, and minimal response times. Addressing these requirements necessitates a precise alignment of system architecture, indexing techniques, and execution strategies to ensure robust and scalable solutions that can also complement hybrid and multi-objective query optimization frameworks.

Maintaining efficient real-time updates to vector indexes presents significant challenges, as traditional static structures lack the flexibility to handle dynamic workloads with frequent updates, deletions, or insertions. Incremental updating methods, such as adaptive indexing strategies, have been developed to minimize computational overhead by targeting only incremental changes rather than full index reconstructions. For instance, dynamic maintenance of compressed representations enhances performance metrics by employing efficient index rebuilding methods like k^2-tree optimization [51] or leveraging incremental graph reorganization techniques [47]. Similarly, snapshot-based indexing asynchronously updates portions of the index, ensuring consistency while optimizing throughput without sacrificing query accuracy in demanding environments [10].

Low-latency query execution benefits significantly from hardware-optimized designs, which leverage advanced accelerators to achieve unprecedented computational efficiency. GPU and FPGA-based frameworks, such as Hypercurves and FANNS, have demonstrated up to 30x improvements in query throughput for high-dimensional similarity

search tasks through techniques like fine-grained task partitioning and workload balancing across CPUs and GPUs [9], [29]. Employing SIMD instruction sets for parallelizing distance computations, such as the vectorized algorithm in SIMD-BP128, further enhances resolution speeds by decoding billions of integers per second [52]. This aligns with broader trends in query optimization, where computation efficiency and throughput under hybrid scenarios are major considerations.

Concurrency management is another critical factor in real-time systems, particularly in distributed environments where maintaining consistency and throughput demands careful trade-offs. Multi-version concurrency control (MVCC), integrated with delta indexing models, supports concurrent read-write workloads without significant service interruptions, providing a balance between performance and accuracy [10]. Moreover, workload-aware query routing and partitioning of vector data have proven effective in improving response times by dynamically allocating resources across hybrid systems [14].

Emerging methods are increasingly focused on optimizing for streaming data dynamics, minimizing latency by adapting to evolving data distributions without requiring complete index reconfigurations. Techniques like Locally-Adaptive Quantization (LVQ) recalibrate vector quantization parameters in real time, enabling superior search performance under rapidly changing workloads [12]. Similarly, advancements in second-tier memory architectures, such as those utilizing CXL-based designs, alleviate disk I/O bottlenecks by achieving near-DRAM-level efficiency for index storage, as highlighted in recent studies [35].

Looking forward, hybrid architectures that integrate hardware-algorithm co-design will likely drive innovations in energy efficiency, scalability, and sub-millisecond latency for billion-scale vector indexes. Cutting-edge technologies, including neuromorphic systems and quantum accelerators, offer the potential to revolutionize real-time query resolution by enabling breakthroughs that address the stringent demands of dynamic, data-intensive applications [9], [40]. As adaptive maintenance, distributed synchronization, and hardware-aware execution frameworks converge, these advancements will form a cornerstone for addressing the computational challenges outlined in hybrid and multi-objective query optimization, facilitating the evolution of next-generation vector database systems.

3.5 Frameworks and Techniques for Multi-Objective Query Optimization

Optimizing query execution in vector database systems often necessitates balancing competing objectives—scalability, accuracy, computational cost, and energy efficiency. This subsection explores frameworks and techniques designed to address these challenges in a unified manner, reflecting the increasingly diverse requirements of modern data applications.

Cost-aware optimization considers trade-offs between accuracy and computational expense, leveraging approximate nearest neighbor search (ANNS) techniques like graph-based methods (e.g., HNSW) or locality-sensitive hashing (LSH). These approaches reduce query latency but introduce configurable trade-offs in retrieval precision [1], [13]. Dynamic query pruning further minimizes computational overhead by adaptively halting similarity computations when pre-defined thresholds are met [53]. However, the design of effective hybrid scoring mechanisms to maximize both precision and efficiency remains a challenge in heterogeneous workloads, especially for hybrid queries combining structured and vector-based predicates [14].

Simultaneously, energy-efficient query execution has emerged as a critical design goal. Techniques such as vector compression play a dual role here, reducing memory footprints while maintaining the computationally intensive distance calculations required in high-dimensional searches [21]. Novel methods like Locally-adaptive Vector Quantization (LVQ) significantly optimize both energy and I/O requirements without substantial accuracy degradation. Hardware-software co-optimization also contributes to energy efficiency, with accelerators like GPUs and TPUs enhancing parallelism while adhering to energy constraints [22].

In the domain of scalability, distributed architectures employing sharding and adaptive routing—often utilized in graph-based systems—represent a core strategy for multi-objective optimization. Systems such as Starling effectively combine in-memory navigation with reordered disk-based structures to maximize processing throughput while maintaining low-latency characteristics for massive datasets [27]. The use of hierarchical balanced clustering further balances index size and limits expensive disk operations in billion-scale vector indices [13].

Context-adaptive optimization techniques represent a burgeoning area of research. These methods adjust query execution strategies dynamically based on workload patterns, user preferences, or domain-specific requirements. For instance, workload-driven partitioning of vector indices demonstrates significant performance gains in hybrid queries by localizing relevant vector groups [14]. Similarly, multi-objective Bayesian frameworks, as exemplified by VDTuner, optimize system performance by automatically navigating large, multi-dimensional parameter spaces without requiring exhaustive manual tuning [39]. These techniques underline the importance of integrating machine learning into system optimization processes.

Despite promising advances, multi-objective query optimization faces several critical challenges. Real-world workloads often involve evolving data distributions and dynamic query types, requiring adaptive, self-updating optimization frameworks. Furthermore, cross-objective trade-offs—such as reconciling energy efficiency with precision in latency-sensitive applications like recommendation engines—necessitate novel algorithmic innovations. Looking forward, emerging paradigms such as federated architectures and quantum-enhanced similarity search hold the potential to expand multi-objective optimization by introducing context-aware execution strategies and fundamentally new resource trade-offs [15].

In conclusion, the development of robust frameworks and techniques for multi-objective query optimization is vital for the sustained evolution of vector database systems. By addressing current limitations and embracing emerging technologies, these approaches can lead to scalable, high-

performance, and environmentally sustainable solutions tailored to the diverse demands of modern applications.

4 Integration with Machine Learning Workflows and Industry Applications

4.1 Vector Databases as Centralized Embedding Repositories

Vector databases have emerged as critical infrastructures for storing, managing, and querying high-dimensional embeddings, which are representations generated by machine learning models from diverse data modalities such as text, images, audio, and video. Their role as centralized repositories facilitates efficient embedding storage, dynamic updates, and retrieval mechanisms, supporting a wide array of data-centric workflows and real-world applications.

Efficient storage is foundational to the functionality of these databases, particularly given the scale and dimensionality of modern datasets. Techniques like vector compression through methods such as Product Quantization or Locally-Adaptive Vector Quantization (LVQ) [12], [21] enable vast reductions in memory footprints while preserving retrieval accuracy. Complementary hardware-optimized structures, such as hierarchical balanced clustering [13], further bolster performance by ensuring that disk or main memory requirements align with operational constraints. These strategies address the exponential growth in embedding space size, allowing efficient long-term maintenance of centralized repositories.

Dynamic embedding updates, critical for evolving machine learning pipelines, present challenges tied to synchronization and versioning. Systems such as FreshDiskANN [50] provide real-time insertion and deletion capabilities, significantly reducing system downtimes associated with index rebuilding. These methods, which rely on optimized update rules combined with graph-based index structures, ensure the database remains up-to-date with incoming vectors. Moreover, explicit integration of version control mechanisms ensures embeddings generated by updated models coexist seamlessly with legacy representations, enabling backward compatibility and enabling AI reproducibility.

Metadata management extends the utility of embedding repositories by enriching vector representations with contextual domain knowledge. For example, in hybrid query execution combining vector similarity and attribute filtering, metadata facilitates precise retrieval, aligning structured and unstructured data seamlessly [14]. Similarly, the synergy of embeddings and metadata-based annotations enhances cross-modal queries by contextualizing numerical vector spaces with semantic meaning, as demonstrated in expanding the scope of multimodal applications [18].

Looking forward, scalable integration with large language models (LLMs) poses an interesting frontier. Embeddings serve as externalized memory structures complementing LLMs, alleviating memory constraints and improving retrieval-augmented generation (RAG) systems [15]. However, ensuring robust latency handling, fault tolerance, and indexing under the shifting distributions of query workloads remains an open challenge [54]. Future research may explore context-specific optimizations, automated index reorganization via learned indexes [26], and tighter integration of multimodal queries to forward industrial applications across domains like healthcare, e-commerce, and personalized AI systems.

As centralized embedding repositories, vector databases are essential not only for storage and retrieval but also for enriching the functionality of machine learning workflows. Their reliability, scalability, and adaptability to emerging trends ensure their pivotal role in unlocking the potential of AI systems, spanning academic research and industry use cases.

4.2 Integration with Machine Learning Pipelines

Vector databases have become integral components of modern machine learning (ML) pipelines, offering robust solutions for efficiently managing the high-dimensional data generated through representation learning. Their interoperability with popular ML frameworks like TensorFlow and PyTorch ensures seamless workflows, where ingestion, storage, and retrieval of embeddings align smoothly with both training and inference pipelines. This subsection investigates the intricate technical integration between vector databases and ML systems, emphasizing key methodologies, associated trade-offs, and emerging frontiers.

One fundamental role of vector databases in ML pipelines is their function as highly efficient embedding repositories, enabling streamlined storage and retrieval across varied machine learning stages. For instance, systems such as Manu [10] implement write-ahead logs that facilitate structured ingestion, ensuring synchronization between the evolving demands of ML models and the underlying storage mechanisms. Furthermore, the use of learned index structures [8], [28] offers adaptive indexing solutions that optimize query performance and storage, significantly enhancing the retrieval precision of high-dimensional embeddings.

Low-latency vector searches are another critical feature of vector databases when embedded within ML pipelines. Applications such as recommender systems, conversational AI, and retrieval-augmented generation (RAG) rely on real-time query execution to provide dynamic, responsive outputs. By leveraging approximate nearest neighbor (ANN) algorithms—such as hierarchical navigable small-world (HNSW) graphs [24] or locality-sensitive hashing (LSH) [6]—vector databases meet the stringent performance benchmarks required by real-time applications, balancing query speed and precision.

In addition to retrieval efficiency, vector databases empower dynamic feature enrichment within ML pipelines. This capability enables embeddings to be extended with auxiliary information, enriching their contextual relevance for downstream tasks. Multimodal embeddings, which integrate representations from different data sources such as text, images, or audio, are particularly noteworthy in facilitating cross-modal searches for sophisticated AI applications [2]. Advances in quantization and vector compression—utilizing methods like Locally-Adaptive Quantization (LVQ) [12], [21]—further optimize the scalability of vector databases. These advancements reduce storage overhead while enabling fast, real-time updates within resource-constrained environments.

However, notable challenges remain. Synchronizing continuously evolving embeddings with distributed or federated deployments of vector databases remains a complex issue, particularly within decentralized machine learning workflows. Additionally, the inherent scalability limitations caused by the curse of dimensionality continue to challenge traditional indexing and querying mechanisms. Future research opportunities lie in addressing these pain points through adaptive consistency models, federated index training frameworks, and tighter integration with emerging tensor-native platforms, as suggested in works like [23]. Incorporating AI-first features, such as auto-tuning of learned indexes or context-aware vector enrichment, may further streamline operations at the intersection of ML and databases.

Finally, a promising avenue involves leveraging vector databases to facilitate hybrid query paradigms in ML-driven workflows. These paradigms integrate structured query mechanisms (e.g., SQL-like filters) with semantic vector-based searches, offering a versatile foundation for solving complex AI challenges [14]. Continued innovation in optimizing these hybrid approaches could open new possibilities, driving broader applicability across diverse domains and reinforcing the indispensable role of vector databases in advancing the state of machine learning systems.

4.3 Multimodal Data Handling and Retrieval

Efficiently managing multimodal data—spanning text, images, audio, and beyond—is a core challenge for modern vector databases, driven by the increasing demand for cross-modal retrieval in domains like multimedia search, recommendation systems, and content understanding. Multimodal data handling leverages high-dimensional vector representations (embeddings) to unify heterogeneous data modalities into a shared mathematical space, optimizing similarity-based retrieval while preserving context and meaning across modalities.

One pivotal approach involves the construction of unified embedding spaces that map distinct data types (e.g., text and images) into a common vector space. Techniques such as contrastive learning and cross-modal alignment have been widely applied, enabling similarity-based queries across modalities. Methods like those found in retrieval-augmented frameworks [12] and graph-based systems [41] demonstrate the utility of such representations in improving retrieval precision. For instance, text embeddings derived from pre-trained language models can be seamlessly aligned with image features extracted from convolutional neural networks, enabling use cases such as visual question answering or caption-based image retrieval.

Cross-modal query execution, a cornerstone of multimodal retrieval, introduces unique challenges. Hybrid indexing structures, such as navigable proximity graph-based systems with structured constraints [41], exemplify strategies for efficient multimodal queries that combine categorical metadata (e.g., labels or tags) with vector-based similarity search. Similarly, systems deploying locality-sensitive hashing in tandem with multimodal metadata annotations [55] enhance retrieval fidelity by reducing the computational overhead associated with comparisons across heterogeneous data sources. These techniques balance trade-offs between query speed and search accuracy, particularly for high-dimensional data, by employing efficient approximate nearest neighbor (ANN) methods.

A critical factor in multimodal data handling is metadata enrichment, which incorporates auxiliary information alongside vector embeddings, enriching contextual understanding during queries. For example, multimodal annotation frameworks integrate hierarchical descriptors for image-text pairs or audio-video collections, improving the precision of cross-modal searches in domains like medical imaging or legal document retrieval [56]. Metadata-aware query structures, such as hybrid query plans [41], further optimize cross-modal searches by dynamically partitioning query evaluation across structured and vectorized components.

Despite advancements, the scalability of multimodal systems remains a pressing problem. High computational demands of dynamic embedding updates, particularly in streaming or real-time contexts, strain storage and retrieval pipelines. Emerging paradigms, such as learned indexing systems for shared clusters across modalities [57], promise to mitigate these challenges by adapting index layouts to shared multimodal distributions. However, trade-offs in storage efficiency versus retrieval accuracy persist [35]. Techniques combining quantization [12] with graph-based strategies [50] show potential for compressing vector data while maintaining high retrieval fidelity.

Future directions include exploring deeper integration between advanced machine learning paradigms and multimodal vector databases. For example, leveraging foundation models like large language models or generative adversarial networks has the potential to redefine cross-modal embeddings by introducing domain-general representational hierarchies. The convergence of quantum computing and vector-based retrieval also highlights transformative opportunities for addressing the computational bottlenecks intrinsic to complex multimodal workloads [28]. Robust benchmarks for evaluating multimodal query systems across diverse datasets will be essential to drive research forward.

In conclusion, vector databases represent a critical enabler for multimodal retrieval, yet balancing scalability, accuracy, and resource efficiency remains complex. Continued innovation in cross-modal embedding techniques, hybrid indexing, and metadata-driven search strategies will play a pivotal role in addressing these challenges and unlocking the full potential of multimodal systems across industries.

4.4 Industry Applications and Use Cases

Vector databases are increasingly recognized as essential infrastructure across industries, powering real-time applications that integrate high-dimensional data retrieval with machine learning (ML) systems. Their primary strength lies in their ability to store, manage, and query vector embeddings that encode complex unstructured data, including textual, visual, and multimodal formats. This subsection highlights prominent industry use cases, focusing on their capabilities, trade-offs, and challenges in practical deployment scenarios.

In e-commerce, recommendation systems heavily rely on vector databases to generate personalized recommendations by leveraging user behavior embeddings. For instance, vec-

torized representations of clickstream data or product features enable real-time similarity searches to identify relevant products, thereby enhancing user satisfaction and conversion rates [1]. By employing approximate nearest neighbor (ANN) techniques, these systems achieve scalability without sacrificing retrieval accuracy. However, technical challenges persist in integrating hybrid queries that combine vector-based similarity with structured filtering constraints, such as price range or category [14].

Semantic search applications in the legal and academic domains utilize vector databases to achieve contextually enhanced document retrieval. Embeddings derived from advanced natural language processing (NLP) models, such as BERT or Sentence Transformers, capture the semantic meaning of extensive text corpora, outperforming traditional keyword-based searches in scenarios requiring high precision [22]. Nevertheless, these implementations face scalability bottlenecks due to the rapid growth of embedding dimensions and the corresponding need for efficient compression methods, such as product quantization or binary encoding [46].

In finance, vector databases enhance fraud detection systems by modeling customer behavior patterns as high-dimensional embeddings. These systems enable real-time identification of anomalous transactions using similarity metrics like cosine or Euclidean distance. For example, hybrid hardware deployments, as seen in systems like Hypercurves, have demonstrated significant query response improvements in time-critical scenarios [29]. However, maintaining recall precision while dynamically updating embeddings in evolving environments remains a critical research challenge.

The healthcare industry leverages vector databases to assist clinical decision-making by querying vectorized representations of patient records, such as electronic health data or radiological scans, to retrieve similar cases for diagnostics and treatment planning [1]. Notably, the use of multimodal embeddings that combine textual clinical notes with medical imaging shows increasing potential [45]. Despite these advances, regulatory compliance, data encryption, and privacy-preserving methods for representation storage present significant obstacles [10].

Across these diverse sectors, vector databases also play a vital role in Retrieval-Augmented Generation (RAG) systems, which augment generative AI by integrating external knowledge retrieval. By using vector embeddings to enhance real-time augmentation of generative outputs, these systems address limitations such as incomplete or outdated model responses [15]. However, RAG workflows often face constraints related to search latency and balancing trade-offs between index size and query speed [12].

To address the growing demands of real-world applications, future advancements in vector databases will likely emphasize scaling to multimodal and hyperdimensional data, improving hybrid query capabilities, and achieving energy-efficient operations in production environments. Promising innovations include storage frameworks such as Delta Tensor [58] and hardware acceleration techniques that co-design algorithms and hardware for high-performance vector search [9]. These advancements will be critical to ensuring that vector databases meet the evolving needs of industries and maintain their pivotal role within modern AI ecosystems.

4.5 Enabling Retrieval-Augmented Generation (RAG) Systems

Retrieval-Augmented Generation (RAG) systems have emerged as a pivotal framework for enhancing generative models, particularly large language models (LLMs), by integrating external knowledge retrieval into their workflows. The core of RAG lies in coupling the generative capabilities of LLMs with high-dimensional vector representations, enabling models to retrieve contextually relevant information from external sources, such as vector databases, which serve as efficient repositories for embeddings. This integration addresses fundamental challenges in generative AI systems, including hallucination, outdated knowledge, and limited context windows.

Vector databases play a critical role in RAG systems by enabling efficient similarity-based retrieval of embeddings. These databases support high-throughput Approximate Nearest Neighbor Search (ANNS) for embeddings generated from various modalities, such as text, images, and audio [10], [13]. Indexing techniques, including hierarchical navigable small-world (HNSW) graphs and locality-sensitive hashing (LSH), underpin these operations, balancing speed and accuracy [13], [21]. For example, integrating vector databases with LLMs allows the models to query domain-specific knowledge, augmenting their generative outputs with accurate and up-to-date information, as demonstrated in industrial applications [10], [27].

The integration of LLMs with vector databases enables RAG systems to overcome traditional limitations of knowledge-restricted language models. By acting as external memory architectures, vector databases mitigate hallucinations—generative inaccuracies common in LLM systems—by anchoring responses to verifiable vector-based information retrieval [15]. This functionality is particularly valuable in dynamic domains such as healthcare and legal research, where the ability to merge structured reasoning with reliable contextual augmentation is critical [2].

Nonetheless, deploying RAG systems introduces trade-offs. Balancing scalability and latency is a notable challenge, particularly when managing embeddings at billion-scale [13]. Additionally, the curse of dimensionality in ANNS methods can degrade retrieval accuracy as vector databases grow in both size and dimensionality [10]. Innovations in vector compression techniques, such as locally adaptive vector quantization (LVQ), address some of these concerns by reducing memory overhead and bandwidth utilization while preserving accuracy [21].

Future developments in RAG systems will likely revolve around refining query optimization and hybrid search mechanisms. For instance, hybrid pipelines that blend vector-based retrieval with structured relational queries are gaining traction for their ability to handle heterogeneous workloads [14]. Advances in workload-aware vector partitioning and multi-query optimization have already demonstrated significant throughput improvements [59]. Furthermore, the adoption of scalable, cloud-native architectures, such as Manu, enhances the elasticity and performance of vector databases in RAG applications [10].

Despite recent progress, open challenges remain. Defining standardized benchmarking methodologies that account for multimodal and cross-modal RAG workflows is still nascent, requiring careful consideration of latency, recall, and relevance metrics [1]. Moreover, increasingly complex application demands in domains like conversational AI and recommendation systems necessitate further optimizations in dynamic index reconfiguration and fault tolerance mechanisms.

In summary, vector databases underpin the transformative potential of Retrieval-Augmented Generation systems by enabling scalable, high-precision embedding retrieval, aligning generative models with external knowledge. Continued innovation in indexing, compression, and hybrid querying strategies promises to elevate RAG's capabilities, paving the way for highly reliable and context-aware AI systems.

4.6 Challenges and Best Practices for Industrial Adoption

The adoption of vector databases in industrial machine learning (ML) workflows has introduced transformative capabilities, yet the journey toward seamless integration into industrial pipelines remains challenging. This subsection delves into these challenges and provides insights into best practices, informed by the latest advancements and empirical studies.

A critical challenge pertains to achieving scalability in real-world deployments. Modern vector databases need to manage billion-scale datasets, which often necessitate distributed architectures for storage and computation. However, enabling low-latency similarity search and efficient update mechanisms within distributed environments is a complex task. Solutions such as sharding and data partitioning play a pivotal role by distributing workloads, though they require sophisticated coordination to minimize cross-node communication overhead. For instance, Manu [10] exemplifies a cloud-native approach that achieves scalability through modularized write components and read-optimized subscribers, leveraging multi-version concurrency control. Despite such innovations, memory-intensive workloads—especially during high-throughput query scenarios—remain a bottleneck. Techniques like delta consistency models and hardware-aware compression strategies, as proposed in [21], mitigate these memory constraints, albeit with trade-offs in computational overhead during queries.

Another persistent issue is the curse of dimensionality, which hinders both the efficiency and precision of similarity searches in high-dimensional vector spaces. Adaptive indexing and approximate methods, including locality-sensitive hashing and navigable proximity graphs, offer practical solutions but can sacrifice accuracy on certain datasets. Hybrid indexing structures designed for multimodal queries, as demonstrated in [41], effectively balance structured and unstructured query requirements. However, such systems often demand substantial computational resources when scaling to industrial-sized datasets. Compression techniques, such as locally adaptive quantization, highlighted in [12], further optimize performance by conserving memory without compromising retrieval fidelity.

Operational challenges frequently emerge as ML pipelines evolve, such as embedding maintenance and retraining, which can introduce "drift" in vector representations over time. Systems like APEX [30] tackle this issue by offering updatable vector indices that integrate smoothly with training workflows, allowing seamless real-time embedding updates without degrading system performance. Additionally, augmenting vector searches with metadata for domain-specific applications, as illustrated in [60], enhances retrieval precision and interpretability. However, incorporating metadata increases dimensionality, complicating the design of efficient index structures.

Cost and operational complexity present another set of challenges in industrial adoption. Automated tuning frameworks such as VDTuner [39] deploy Bayesian optimization algorithms to systematically adjust parameters, striking an optimal balance between recall and latency while adhering to resource constraints. Moreover, leveraging hardware accelerators like GPUs or FPGAs, as discussed in [9], significantly reduces computational latency, enhancing overall performance. Nonetheless, adopting such specialized hardware entails high upfront costs and requires adapting workflows to match hardware capabilities.

As vector database technology matures, its co-evolution with ML pipelines is imperative. Future designs should emphasize dynamic indexing, adaptive optimizations for hybrid query execution, and energy-efficient hardware-software co-design. Collaboration between academia and industry proves essential, particularly in benchmarking novel configuration paradigms that address workload-specific and multimodal challenges. With advancements in both algorithms and system architectures, vector databases are poised to become indispensable components of industrial ML systems, driving scalable, reliable, and application-oriented solutions for next-generation use cases.

5 Performance Benchmarking and Evaluation Metrics

5.1 Benchmarking Frameworks and Experimental Design

The benchmarking of vector database management systems (VDBMSs) is critical for rigorous evaluation, fostering comparisons, and guiding advancements in performance, scalability, and usability. This subsection explores the frameworks and experimental designs that underpin standardized benchmarking efforts, focusing on realistic workload modeling, configuration diversity, and dynamic operational scenarios.

Benchmarking frameworks for VDBMSs typically rely on both synthetic and real-world datasets to evaluate performance across a wide range of use cases. Synthetic datasets, which are algorithmically generated, offer precise control over data dimensionality, volume, and distribution. They are pivotal for testing specific aspects, such as resilience against the "curse of dimensionality" or efficacy in dynamic environments [4], [45]. Real-world datasets, meanwhile, capture the inherent complexities found in operational domains—e.g., multimedia, healthcare, and high-dimensional textual embeddings—ensuring that evaluations align with real application demands [2], [61]. Their integration within

benchmarks helps address domain-specific dependencies, including vector sparsity, heterogeneity, and evolving data distributions.

Experimental workload design is a critical component. Workloads typically include combinations of nearest neighbor queries, hybrid similarity searches, and multimodal queries. Query distributions are modeled to reflect practical scenarios, ranging from uniform random distributions to skewed workloads that mimic user behavior in domains like recommendation systems and image-based e-commerce [14], [61]. Hybrid workloads, which combine structured SQL-like queries with vector similarity searches, push the boundaries of query engines, emphasizing challenges related to query planning and optimization [18].

Existing benchmarking frameworks, such as LDBC and IoTDB-Benchmark, provide a foundation for standardized testing but often lack dedicated support for high-dimensional, vector-focused use cases [62]. Recent efforts propose more tailored tools, such as FreshDiskANN, emphasizing metrics like recall-at-top-k, query latency, throughput, and scalability [50]. These tools often include mechanisms for stress testing VDBMSs under dynamic conditions, such as real-time updates, deletions, or high-concurrency environments, which are fundamental for evaluating performance in streaming or large-scale deployments [10], [50].

Frameworks increasingly highlight the need for adaptive benchmarking techniques. For instance, dynamic benchmarks can assess how VDBMSs evolve under continuous workloads, analyzing behaviors during index updates or retraining processes for embeddings [50], [54]. These dynamic scenarios are particularly critical given that vector embeddings often emerge from machine learning models subject to iterative improvements [19].

Emerging trends include integrating multimodal benchmarks that assess performance across text, image, and audio embeddings, as well as frameworks for hybrid query optimization combining structured and unstructured workloads [8], [63]. Another frontier involves energy-aware benchmarks, reflecting the environmental costs of large-scale vector operations [9]. Future efforts must also address privacy-preserving benchmarking, where encrypted vectors or differential privacy introduce constraints on both data and computational models [28].

In summary, robust benchmarking frameworks and experimental designs remain foundational for evaluating and advancing the state of VDBMSs. Evolving benchmarks must not only account for performance metrics but also adapt dynamically to reflect the evolving operational contexts and user requirements of these systems.

5.2 Evaluation Metrics for Vector Databases

Evaluation metrics for vector databases are critical for assessing their performance, reliability, scalability, and suitability across diverse application scenarios. These metrics provide quantifiable insights into system capabilities, enabling researchers, practitioners, and developers to gauge how effectively a vector database supports tasks such as nearest neighbor search, multimodal querying, and hybrid workloads. As discussed in the benchmarking subsection, the selection of evaluation metrics plays a pivotal role in characterizing system strengths and limitations under varying use cases.

Key performance metrics include *query accuracy*, *latency*, and *throughput*. Accuracy metrics—such as recall, precision, and top-k accuracy—are particularly significant in approximate nearest neighbor search (ANNS). Recall, which measures the proportion of relevant results retrieved, is a cornerstone metric for evaluating systems that trade speed for precision. Techniques like product quantization [46] and graph-based approaches [24] showcase the scalability of such systems but often require nuanced hyper-parameter tuning to maintain optimal recall without significant degradation in latency. Similarly, precision, especially in the context of high-dimensional spaces, is indispensable for reducing noise in the results, ensuring a balanced trade-off between recall and unwanted retrievals when working with noisy or sparse data distributions.

Scalability metrics go hand in hand with performance assessments, as modern vector databases must support billions of vectors while maintaining acceptable response times. These metrics often evaluate the effects of increasing *dataset size* or *dimensionality* on key performance indicators such as response latency or memory consumption. Advanced memory-efficient data structures, such as those employing vector compression through quantization [12], [21], have emerged to balance storage overhead and retrieval performance. Notably, approaches like Locally-Adaptive Vector Quantization (LVQ) [12] highlight the adaptability required in dynamic streaming scenarios, where dataset distributions and sizes evolve continuously.

Resource utilization metrics—spanning CPU, GPU, memory, and I/O usage—offer a lens into operational efficiency and the interplay of hardware with algorithmic frameworks. Systems such as Faiss [5] and Starling [27] exemplify the importance of hardware-aware optimizations in achieving substantial performance gains. For instance, Faiss leverages SIMD instructions for parallelized similarity computations, whereas Starling's graph reordering techniques minimize disk latency for graph-based, disk-resident indexes in high-dimensional search tasks. Such innovations directly address real-world constraints in both memory-bound and disk-bound environments.

Multimodal query evaluation represents another pivotal dimension, assessing a system's ability to seamlessly handle combinations of vector similarity searches and traditional attribute-based filtering. Hybrid workload scenarios, where unstructured vector-based queries intersect with structured attribute constraints, challenge databases to optimize across diverse indexing paradigms. For example, by integrating vector similarity search with relational data techniques [14], these systems showcase their capacity for complex query execution, particularly in applications like recommendation engines and semantic retrieval.

Sustainability metrics, an emerging evaluation focus, reflect the growing emphasis on energy efficiency and environmental responsibility within large-scale vector search deployments. Metrics such as energy-per-query and carbon footprint highlight the broader impact of scaling database systems, motivating the adoption of green AI practices like adaptive query strategies and energy-aware optimizations.

Finally, the future trajectory of evaluation metrics is expected to align more closely with evolving data and workload dynamics, emphasizing real-time adaptability and workload-awareness. Innovative strategies such as dynamic recall metrics and learned benchmarking frameworks [8] promise to expand on existing methodologies by accommodating shifts in data distribution or federated query scenarios. Such adaptive evaluation techniques are essential for capturing the performance of modern VDBMSs under continuously evolving operational demands.

By synthesizing metrics across accuracy, scalability, resource utilization, and real-time adaptability, this domain establishes comprehensive frameworks for reliably comparing vector database systems. These foundational evaluation strategies not only complement the benchmarking methodologies discussed earlier but also pave the way for the comparative studies of open-source and commercial systems in the following section, ensuring coherent progress across research and industrial innovation.

5.3 Comparative Evaluation of Existing Systems and Approaches

A comparative evaluation of existing vector database management systems (VDBMSs) provides critical insights into the spectrum of design trade-offs, implementation approaches, and performance bottlenecks. This subsection explores both open-source and commercial systems, employing established benchmarking methodologies and metrics to systematically analyze their architectures, scalability, efficiency, and application contexts. Primary focus is placed on indexing techniques, similarity search paradigms, and optimization strategies, as these fundamentally impact query performance, throughput, and resource utilization.

Prominent open-source systems, such as Faiss [5], Annoy, and Milvus, have emerged as benchmarks in scalability and adaptability. Faiss emphasizes GPU support, leveraging hardware acceleration for high-dimensional vector retrieval, and implements optimized techniques such as product quantization for memory-efficient storage while achieving substantial speed gains in large-scale benchmarks [5]. Meanwhile, Milvus integrates hierarchical navigable small-world (HNSW) graphs with distributed query processing, enhancing both low-latency performance and scalability over billion-scale datasets [13]. However, these systems often prioritize approximate nearest neighbor (ANN) search at the expense of guaranteed precision, posing challenges when stringent accuracy requirements prevail, particularly in applications involving hybrid vector-attribute queries [41].

Commercial systems exhibit broader functionality tailored to enterprise use cases. Pinecone and Weaviate, for example, focus on end-to-end workflows by combining vector indexing with metadata-based filtering, enabling seamless hybrid query execution. Pinecone's disk-aware architecture optimizes SSD usage to overcome memory constraints in large-scale settings, while maintaining query time predictability. These systems excel in operational simplicity and integration with enterprise pipelines but may lag in performance tuning options compared to OpenAI-optimized instances of Lucene [49], which highlight the potential of adapting traditional full-text systems for vector-centric applications.

Recent advances in learned indexes further redefine the trade-offs in storage and retrieval across systems. Techniques such as LIPP [64] and ALEX [65] utilize machine learning to predict key positions, significantly reducing index size while maintaining high lookup speeds. However, such approaches face limitations when handling dynamic updates or highly skewed data distributions, as observed in adaptive experiments [64]. Additionally, approaches like LIMS [26], which combine clustering strategies and learned models for exact search, underline the potential of learned systems to transition ANN workloads into deterministic similarity search scenarios, albeit with increased preprocessing complexity [26].

Graph-based systems, exemplified by HNSW and novel designs such as NSSG [7], excel in memory-disk hybrid scenarios by balancing graph sparsity and search performance. Meanwhile, disk-centric implementations like SPANN leverage hierarchical clustering to reduce I/O overheads, achieving near-real-time performance even under constrained memory conditions [13].

Future benchmarking must consider hybrid workloads more comprehensively, integrating evaluation of structured and vector data queries into unified metrics. Moreover, benchmarking frameworks need revisions to encompass scenarios involving real-time updates, multimodal retrieval, and federated systems, as identified in works like FreshDiskANN [50]. Addressing these open challenges will aid in defining the next generation of vector databases, bridging the gap between specialized performance and general-purpose applicability.

5.4 Domain-Specific Benchmarks Across Industries

Benchmarks tailored to specific industries play an essential role in evaluating the practical efficacy and robustness of vector database management systems (VDBMSs). Unlike general-purpose benchmarks, these domain-specific evaluations are closely aligned with the unique data characteristics, query patterns, and operational constraints of their respective industries, thereby offering more actionable and targeted insights. This subsection explores critical benchmarks across healthcare, e-commerce, multimedia retrieval, and natural language processing (NLP), assessing their design principles, trade-offs, and ongoing challenges, while situating these discussions in the broader context of VDBMS optimization and application-specific demands.

In healthcare, VDBMS benchmarks address stringent requirements for accuracy and reliability in retrieving high-dimensional data, such as electronic health records (EHRs), genomic datasets, and medical imagery. Given the high stakes of clinical decision-making, evaluation frameworks in this domain often emphasize metrics like recall and precision over query latency, ensuring that the systems support robust and accurate retrieval. Techniques such as compressed indexing and specialized optimization for medical trajectory data or genomic sequences [42], [66] are of particular importance. These benchmarks must also accommodate the multimodal nature of healthcare datasets, ensuring that both structured metadata and vector embed-

dings are effectively integrated to deliver meaningful search results [45].

E-commerce applications, on the other hand, emphasize dynamic personalization and recommendation systems, necessitating benchmarks that highlight response latency, personalization accuracy, and user experience metrics such as result diversity and novelty. For this domain, hybrid query execution involving both vector similarity and attribute-based filters is heavily benchmarked. Optimized space-partitioning strategies [67] provide essential insights, balancing computational overhead against recall to deliver near-instantaneous and tailored recommendations. These benchmarks, therefore, highlight the need for VDBMSs to perform seamless real-time queries while managing large-scale datasets.

In multimedia retrieval tasks, such as those involving video and image search, benchmarks encounter the challenge of evaluating systems against the high computational costs of extracting and comparing embeddings from large datasets. Approaches like Locally-Adaptive Quantization [12] and graph-based indexing schemes [21] enable testing under extreme scalability scenarios while maintaining low-latency requirements. Moreover, this domain underscores the importance of striking optimal trade-offs between storage efficiency and retrieval accuracy, making these attributes central to benchmark evaluations.

Finally, in the field of natural language processing (NLP) and chatbot applications, benchmarks often focus on hybrid queries integrating relational attributes with language model-derived vector embeddings. Retrieval-Augmented Generation (RAG) systems, which enhance large language model (LLM) outputs using vector databases, present unique benchmarking challenges centered on optimizing precision, response fluency, and robustness to concept drift in evolving datasets [12]. These benchmarks emphasize testing VDBMSs' ability to address the dynamic and unpredictable nature of real-world language data.

Despite advancements, the development of unified benchmarking frameworks that balance diverse trade-offs—such as latency, memory consumption, and accuracy—remains a critical challenge across industries. Furthermore, future benchmarks must increasingly incorporate real-world constraints, including energy efficiency, fault tolerance in dynamic workloads [68], [69], and data privacy in industries with sensitive data requirements. Closing these gaps will require cross-domain benchmark suites capable of providing holistic evaluations and inspiring innovations that transcend specific industry boundaries. By addressing these needs, benchmarks can continue to evolve alongside VDBMS technologies, supporting their optimization for diverse and emerging workloads.

5.5 Domain-Specific Optimization and Challenges

The optimization of vector database management systems (VDBMSs) is highly domain-dependent, as different industries impose varying demands on accuracy, performance, scalability, and resource efficiency. Addressing these challenges necessitates a nuanced understanding of use case-specific constraints and optimization techniques. This subsection explores key challenges and innovations in tailoring VDBMSs for diverse real-world applications while balancing trade-offs inherent to these optimizations.

One of the foremost challenges lies in achieving efficient approximate nearest neighbor (ANN) search within high-dimensional vector spaces, particularly under domain-specific accuracy and latency constraints. The curse of dimensionality exacerbates the computational expense of similarity search, affecting both memory and query efficiency. Optimized indexing solutions such as memory-disk hybrid approaches [13], and graph-augmented search frameworks with compression techniques like locally-adaptive quantization [21], have demonstrated that indexing architectures tailored to specific query workloads can significantly improve both speed and precision. However, these often encounter bottlenecks when processing hybrid queries that combine structured filters with vector similarity predicates, as seen in knowledge graph applications [14].

Domain-specific scenarios, such as multimedia search or recommendation systems, introduce additional trade-offs between resource usage, scalability, and retrieval accuracy. In domains where frequent real-time updates are required, maintaining consistent, low-latency index updates becomes a priority. For example, delta-based indexing mechanisms that minimize full index rebuilds during state changes are increasingly adopted [70]. Conversely, for workloads dominated by large-scale, infrequently updated datasets, leveraging disk-resident index frameworks like Starling [27] achieves immense scalability with reduced memory requirements, though at the cost of higher query response times for interactive applications.

Energy efficiency has also emerged as a critical optimization axis, especially for resource-intensive deployments such as dynamic multimedia or edge AI workloads. Techniques such as hardware-aware compression [10], reduction of redundant index structures, and leveraging GPUs for efficient vector operations [71] highlight the potential for co-optimizing hardware and software layers in energy-sensitive use cases. However, evaluating these strategies in practice is hampered by a lack of standardized benchmarks that holistically measure energy consumption alongside traditional metrics such as recall and throughput [62].

Looking ahead, cross-domain innovations present exciting opportunities for enhancing VDBMS capabilities. Hybrid indexing strategies that intelligently fuse graph-based and learned partitioning mechanisms [72] hold promise for adapting systems to evolving query distributions. The development of dynamic, workload-aware optimization frameworks such as VDTuner [39] emphasizes the increasing role of automation in fine-tuning system configurations across varying deployment contexts. Nevertheless, challenges such as ensuring fairness in multimodal query optimization, mitigating energy trade-offs, and accommodating out-of-distribution (OOD) query scenarios [54] underline the need for continued innovation in tailoring vector databases for highly specific, yet interconnected, real-world demands.

In summary, while substantial progress has been made in aligning optimization strategies with the unique demands of specific domains, the interplay of scalability, efficiency, and domain-specific constraints remains a fertile ground for future research. Addressing these open challenges will

ensure VDBMSs continue to meet the evolving demands of applications leveraging high-dimensional vector data.

5.6 Future Directions in Benchmarking and Industry Standards

The evolution of performance benchmarking in vector database management systems (VDBMSs) is pivotal in addressing emerging trends and unresolved challenges, helping to establish robust and cohesive industry standards. While traditional benchmarking frameworks have been effective in evaluating fundamental metrics such as query latency and recall [1], the increasing complexity and diversity of modern applications demand more sophisticated methodologies tailored to real-world operational scenarios.

A key direction in benchmarking innovation is expanding beyond unimodal datasets to more realistic, multimodal benchmarks capable of capturing hybrid data types, such as text, image, and video embeddings. Existing benchmarks like ParIS+ [73] and ULISSE [74] excel in optimizing performance for specific query workloads but remain limited to unimodal data. Developing benchmarks that integrate diverse, multimodal datasets for cross-modal retrieval tasks could offer a more comprehensive evaluation framework, enabling vector systems to be tested under the complex and varied conditions prevalent in real-world use cases [1].

Equally critical is the need to incorporate privacy and security dimensions into benchmarking frameworks. As vector embeddings frequently encode sensitive user or organizational information, robust evaluation methodologies must address the trade-offs between retrieval accuracy and security mechanisms. Encrypted vector datasets and privacy-preserving algorithms should be integral to these advanced benchmarks. Solutions leveraging secure storage architectures, such as those discussed in [75], provide a foundation for assessing both performance and security.

In federated and decentralized architectures, benchmarking introduces additional complexities due to diverse latency, storage, and resource capabilities across distributed nodes. Systems designed for such settings need benchmarks that reflect the challenges of heterogeneity while maintaining reproducibility. Techniques like workload-aware data partitioning [14] could guide the development of standards that adequately address the scalability and performance trade-offs unique to federated environments. Notably, systems such as Starling [27] highlight the importance of benchmarking tools that can effectively evaluate performance under distributed configurations.

Machine learning-driven methodologies present transformative possibilities for dynamic benchmarking. By integrating adaptive testing frameworks, such as those leveraging reinforcement learning or Bayesian optimization (e.g., VDTuner [39]), benchmarks can evolve in response to observed workload characteristics. Such an approach would not only increase the relevance and precision of evaluations but also reduce overhead, offering customized insights for diverse deployment scenarios.

Lastly, fostering community-driven, open-source initiatives can bring consistency and collaboration to benchmarking practices. Similar to the collaborative advancements achieved through the Faiss library [5], unified efforts across stakeholders could standardize baseline metrics, encourage transparency, and accelerate innovation within the VDBMS ecosystem.

Looking ahead, the development of holistic benchmarking frameworks equipped to evaluate multimodal data, privacy-enhancing mechanisms, distributed environments, and dynamic workload optimizations will serve as the cornerstone for reliable, scalable, and secure VDBMS solutions. These initiatives must also provide ample flexibility to accommodate future technological shifts, such as quantum computing and neuromorphic architectures, ensuring the perpetual relevance of vector database management systems in emerging domains [38].

6 Challenges, Emerging Trends, and Directions

6.1 Scalability Challenges and Strategies for High-Dimensional Data

The explosive growth of high-dimensional data has created significant scalability challenges for vector database management systems (VDBMSs). Handling massive datasets in dynamic and frequently updated environments demands solutions that optimize storage, accelerate similarity search, and ensure fault tolerance while minimizing computational overhead. Here, we analyze the key aspects of these challenges and strategies to address them, leveraging insights from recent literature.

One of the principal challenges stems from the "curse of dimensionality," where the computational complexity of similarity search increases exponentially with the number of dimensions. Traditional exhaustive techniques for exact nearest neighbor (NN) search falter in large-scale settings due to prohibitive latency and inefficiencies in high-dimensional spaces [4]. Approximate Nearest Neighbor Search (ANNS) techniques, such as hierarchical graph-based indexing [7] and pruning distance-space graphs [50], have emerged as effective solutions, offering trade-offs between precision and query speed. However, strategies like using highly sparse data structures or quantized embeddings, while efficient, introduce accuracy compromises that must be tailored to specific application requirements [12].

Managing rapidly growing datasets poses distinct scalability hurdles. Distributed architectures—focusing on partitioning data across multiple nodes through techniques like dynamic sharding and proximity-aware clustering [13]—help mitigate these challenges by dividing workloads and minimizing cross-node traffic. Elastic scaling mechanisms, exemplified in systems like Manu, dynamically balance resources in response to fluctuating workloads, ensuring both high availability and computational efficiency [10]. Meanwhile, advances in hybrid memory-disk systems have demonstrated scalable performance for billion-sized datasets by effectively leveraging hardware optimizations [27].

Dynamic updates to continuously evolving vector databases introduce additional bottlenecks. Incremental indexing algorithms, such as the real-time update capability of FreshDiskANN, address latency issues by directly incorporating new data into the index without recomputation [50]. Furthermore, redesigning indexing strategies to adapt to

query distributions and dataset shifts—illustrated by OOD-DiskANN's workload-aware methodology—improves system robustness, especially in out-of-distribution workloads [54].

Optimized compression techniques, such as Locally-Adaptive Vector Quantization (LVQ), offer another critical solution for large-scale scenarios. By reducing the storage footprint while preserving accuracy, LVQ advances both search performance and space efficiency [12], [21]. These methods, combined with co-designs integrating hardware accelerators like GPUs and FPGAs, further reduce computational costs in high-throughput environments [9].

Despite these advancements, significant challenges persist, particularly in handling multimodal datasets, maintaining consistency under rapid updates, and addressing emerging privacy-preserving requirements. Future investigations into decentralized architectures, real-time adaptive indexing, and novel benchmarking frameworks for scalability testing are critical [1]. Establishing standardized evaluation methodologies for scalability, especially under dynamically changing workloads, could drive innovations that bridge gaps across scalability, accuracy, and efficiency trade-offs. These advancements represent vital directions for ensuring scalable, high-performance VDBMSs capable of meeting diverse, rapidly evolving real-world demands.

6.2 Privacy and Security in Vector Database Management Systems

The integration of high-dimensional vector embeddings into database management systems introduces a complex set of privacy and security challenges, particularly as these embeddings often encode sensitive or identifiable information. For instance, embeddings derived from personal data — such as facial recognition vectors or behavioral patterns — carry the risk of privacy breaches if improperly managed or accessed, thereby heightening the need for robust protections. This subsection examines these critical challenges and evaluates emerging strategies to mitigate them, focusing on the secure management of embeddings, resilience against adversarial threats, privacy-preserving methodologies, and regulatory compliance, all within the broader context of maintaining system efficiency and usability.

A fundamental concern lies in the secure storage and transmission of vector embeddings, where safeguarding against unauthorized access or leakage is paramount. Advanced encryption techniques adapted for high-dimensional embeddings, such as homomorphic encryption and secure multiparty computation, have shown significant promise in protecting sensitive data while enabling similarity operations. For example, when combined with developments in compressed data structures, such encryption methods can address trade-offs between storage efficiency and access security [76]. Specialized solutions like vectorized embedding encryption further enhance confidentiality by enabling encrypted similarity computations, eliminating the exposure of raw embeddings during operations [42]. However, these methods often introduce computational overhead, which underscores the need for optimization to achieve practical, scalable deployment.

In addition to secure storage, vector embeddings are vulnerable to adversarial attacks, where small perturbations in the data can disrupt search outcomes or expose sensitive information through reverse-engineering. Adversaries may exploit these weaknesses to induce significant deviations in similarity results or to infer private details about underlying data. Strategies such as adversarial training — which incorporates perturbed embeddings into the training process — and geometric techniques like boundary-based defenses are actively being explored to bolster robustness [77]. However, these defenses currently face challenges in scaling to large and dynamic datasets, highlighting a critical area for further innovation.

Privacy-preserving methodologies are becoming increasingly integral to vector database systems, particularly with the advent of techniques like federated learning. These approaches allow collaborative learning across distributed datasets without requiring centralization, thereby significantly reducing privacy risks [78]. Furthermore, the adoption of differential privacy mechanisms ensures that individual embeddings cannot be reverse-engineered during similarity queries, preserving anonymity while maintaining the utility of the system [79]. These innovations represent significant advancements toward safeguarding privacy while enabling powerful query capabilities.

Regulatory compliance and ethical considerations further compound the complexity of embedding management in vector databases. Legal frameworks such as the GDPR and CCPA mandate transparency, data minimization, and ensuring user consent, imposing stringent requirements on systems handling sensitive vectorized data. Key features like auditability, explainability, and user-centric consent controls must be incorporated to meet these obligations [80]. However, the absence of standardized practices for embedding representation and similarity search hinders efforts to align systems with diverse regulatory landscapes, further complicating compliance efforts.

Recent technological trends provide opportunities to address some of these challenges. Blockchain, for instance, is gaining attention as a means of ensuring auditable, tamper-proof storage for vector data, with cryptographic guarantees for access integrity. Similarly, the integration of quantum-safe cryptographic techniques offers a compelling pathway for future-proofing VDBMS designs against potential risks posed by quantum computing advancements.

Looking ahead, research must prioritize the development of scalable privacy-preserving and adversarial-resistant methodologies that are adaptable to the dynamic nature of real-world workloads. Establishing standardized security benchmarks and context-specific testing frameworks will be crucial for evaluating and improving the protective capabilities of these systems. Ultimately, achieving a balance between privacy, performance, and scalability will be vital to fostering trust in vector database management systems as they become integral to increasingly complex and sensitive applications.

6.3 Energy Efficiency and Environmental Sustainability

The rise of vector database management systems (VDBMSs) has coincided with an era of staggering data expansion and embedding-heavy workloads, creating significant computational and environmental challenges due to their high

energy consumption. As these systems scale to serve billions of vectors with low-latency requirements and precision-intensive algorithms, energy usage has emerged as a critical concern. This subsection delves into the environmental impacts of these systems, evaluates innovative energy-efficient strategies, and highlights future research directions.

VDBMS operations, particularly indexing, similarity search, and hybrid query execution, typically involve computationally heavy processes consuming notable energy. For example, some graph-based proximity search methods, while achieving state-of-the-art results, face high indexing complexity, especially when scaling to billion-scale datasets [13]. Similarly, compression techniques such as vector quantization perform memory compression at the risk of intensive compute cycles during retrieval [21]. Increasing reliance on hardware accelerators like GPUs and SSD-based storage further elevates the energy consumption during large-scale similarity searches [27].

Energy-efficient strategies have been actively explored to reconcile performance and environmental sustainability. Techniques like pruning and early stopping, used to strategically reduce vector comparisons, have shown promise in minimizing unnecessary computations [21]. Compression methods such as Locally-Adaptive Vector Quantization (LVQ) reduce data representation sizes while simultaneously limiting hardware bandwidth overhead, achieving lower energy footprints alongside computational gains [12]. Furthermore, emerging SSD-hybrid indexing structures such as SPANN and OOD-DiskANN demonstrate up to 5–10x improvements in disk I/O efficiency, suggesting scalable designs for greener architectures [13], [54].

Hardware-software co-design offers another promising avenue. Persistent memory technologies, such as those integrated into APEX, reduce power consumption by eliminating the high energy costs of traditional on-disk and DRAM-dependent approaches, achieving up to 15x improvements over conventional indexes while maintaining ultra-low latencies [30]. Additionally, frameworks such as FreshDiskANN demonstrate that incremental update mechanisms in graph-based systems can help shift away from periodic re-indexing, yielding significant reductions in energy consumption during dynamic workloads [50].

Lifecycle carbon footprint evaluation frameworks, inspired by broader Green AI initiatives, need to be adapted to measure and optimize the holistic energy consumption of VDBMS pipelines [35]. For instance, integrating renewable energy sources into data center operations and adopting dynamic query scheduling based on energy price fluctuations could bring additional sustainability gains.

However, there are challenges to achieving sustainable VDBMSs. Trade-offs exist between energy efficiency and system performance, particularly in high-dimensional nearest neighbor searches, which are exponentially complex [81]. Moreover, energy-efficient designs must preserve system precision and recall metrics, requiring novel adaptability in index pruning and model-control techniques [65]. Future vectors of research include exploring new cooling technologies, leveraging neuromorphic and quantum computing for energy-efficient computations, and fine-tuning hybrid storage designs to optimize per-query energy costs.

In conclusion, achieving energy-efficient vector database systems demands a multi-faceted approach, integrating technical optimizations, hardware innovations, and lifecycle-aware sustainability frameworks. As VDBMSs play a central role in powering AI-driven applications, reconciling their computational intensity with environmental responsibility will be pivotal for their broad-scale adoption and long-term viability.

6.4 Advancing Multimodal and Hybrid Query Capabilities

The rise of multimodal and hybrid query capabilities in vector database systems marks a pivotal evolution, bridging the increasing complexity of real-world data retrieval demands with the sophisticated infrastructures needed to support them. Multimodal queries, capable of processing heterogeneous data such as text, images, and audio, enable richer and more contextual information retrieval. Hybrid query paradigms, on the other hand, merge structured relational data with unstructured vector-based data, creating seamless integrations vital for advanced applications like cross-modal search, interactive recommendation systems, and AI-driven analytics pipelines. These capabilities exemplify the expanding role of VDBMSs in managing increasingly diverse and interconnected data ecosystems.

Multimodal query execution demands cohesive representations for diverse data modalities, necessitating unified embedding frameworks that can accommodate varying data types. Techniques such as shared embedding spaces leverage vector symbolic architectures, unifying modalities within a common geometric framework to ensure compatibility [19]. Similarly, systems employing hierarchical or layered representations of multimodal data achieve notable compression benefits, which are critical for managing complex datasets while preserving query fidelity across varying formats [82]. However, achieving robust alignment across disparate modalities remains a substantial challenge, particularly when embeddings are generated by domain-specific models with variable data distributions.

Hybrid query processing serves as a critical bridge between structured and unstructured data paradigms. By integrating vector similarity searches with traditional predicate-based filter mechanisms, systems can efficiently handle hybrid workloads that encompass both dense feature vectors and sparse relational attributes. For instance, HQI optimizes hybrid workloads by utilizing workload-aware indexing strategies, which boost throughput while cutting operational costs in batch query settings [14]. Fusion-based methodologies, combining dense vector embeddings with sparse attribute-based indexes, further enrich query semantics without sacrificing scalability [2]. Nevertheless, hybrid indexing systems, such as graph-relational combined methods, often face trade-offs in memory efficiency and query response times, signaling an ongoing need for optimization.

The real-time execution of multimodal and hybrid queries introduces additional layers of complexity. Addressing these challenges demands adaptive storage solutions like dynamic quantization, which efficiently handle evolving datasets, as well as latency-aware retrieval algorithms optimized for responsiveness under dynamic conditions [5], [12]. Moreover, integrating machine learning models

into query optimization strategies presents opportunities to refine performance in dynamic and unpredictable environments [10].

To fully capitalize on the transformative potential of these query paradigms, system-level co-designs are emerging as an essential area of innovation. For example, coupling efficient indexing mechanisms with hardware accelerators, such as GPUs or TPUs tailored for hybrid search, has demonstrated significant improvements in system throughput and scalability [9]. Furthermore, deeper integration of generative AI models within multimodal retrieval workflows, as seen in Retrieval-Augmented Generation (RAG) systems, could further enhance the expressiveness and accuracy of query results [12].

In conclusion, the advancement of multimodal and hybrid query capabilities not only underscores technical ingenuity but also reflects the growing adaptability of VDBMSs to diverse industrial and scientific applications. However, realizing the full potential of these innovations will require rigorous benchmarking practices and the development of standardized hybrid query evaluation metrics. As the field evolves, addressing these challenges will expand the versatility of vector database systems, positioning them as indispensable tools across a broad spectrum of use cases and industries.

6.5 Integration with Revolutionary Technologies

The integration of vector database management systems (VDBMSs) with revolutionary technologies, such as quantum computing, blockchain, and neuromorphic systems, holds immense potential for advancing their capabilities and addressing the unique challenges of high-dimensional data management. These technologies, each with distinct characteristics, offer transformative opportunities for VDBMS functionality, scalability, and security.

Quantum computing emerges as a compelling avenue for accelerating similarity search, especially in large-scale, high-dimensional datasets. By leveraging quantum algorithms like Grover's search, which provides quadratic speedup for unstructured search problems, VDBMSs have the potential to significantly reduce query processing times in approximate nearest neighbor search (ANNS). Recent studies suggest that quantum-inspired optimizations could further enhance classical systems by emulating quantum computational paradigms to improve data partitioning and search efficiencies [83]. However, practical implementation in VDBMSs faces challenges, including the development of hybrid algorithms to ensure compatibility with existing classical frameworks and the inherent probabilistic nature of quantum outputs, which can complicate result determinism in precision-critical scenarios.

Simultaneously, blockchain technology offers innovative pathways to address data security and provenance concerns within VDBMSs. Embedding blockchain-enabled audit trails can ensure that the creation, modification, and usage of vector embeddings remain tamper-evident and verifiable. This is particularly crucial for federated and decentralized systems managing privacy-sensitive information [84], [85]. The challenge lies in integrating blockchain's computational overhead with the low-latency requirements of VDBMSs, necessitating hybrid approaches, such as off-chain indexing for routine queries while reserving blockchain validation for critical state transitions. Furthermore, blockchain could enable decentralized marketplaces for sharing pre-trained embeddings among disparate organizations, unlocking new collaboration paradigms.

Neuromorphic computing presents another revolutionary avenue for VDBMS acceleration by mimicking biological neural networks, enabling energy-efficient and densely parallelized computation. Neuromorphic architectures have shown promise in vector operations, particularly matrix-vector multiplications integral to similarity searches [40]. Their use can achieve drastic reductions in power consumption and execution latency compared to traditional hardware accelerators like GPUs and TPUs. However, the integration of neuromorphic systems requires rethinking foundational data structures and algorithms, ensuring compatibility with event-driven computational models native to these architectures.

Looking toward the future, effective utilization of these technologies demands interdisciplinary collaboration. Hybrid designs, leveraging quantum preprocessing to reduce search space, blockchain to ensure secure storage, and neuromorphic systems for real-time querying, could redefine the possibilities of VDBMSs. Research must address interoperability among these technologies while ensuring scalability and cost-efficiency in production environments. Such endeavors could usher in a new era of VDBMS innovation, enabling unprecedented performance and applicability across diverse domains.

6.6 Open Challenges and Future Research Directions

Vector database management systems (VDBMS) represent a cornerstone for meeting the growing demands of data-driven applications, yet a range of unresolved challenges and research opportunities point to a promising future. This subsection delves into critical hurdles and potential directions that could steer the development of VDBMS toward greater efficiency and capability.

One major challenge is the absence of standardization in VDBMS architecture and evaluation. Current systems employ diverse indexing schemes, storage methodologies, and query optimization strategies, making it difficult to objectively assess their performance across both real-world and experimental scenarios. While some initiatives have advanced this cause [1], [45], there remains a need for universally accepted metrics to capture aspects such as scalability, hybrid query performance, and energy efficiency. For example, benchmarking efforts frequently emphasize isolated components like approximate nearest neighbor search (ANNS) but often neglect holistic assessments of end-to-end system behavior under varied workloads [1], [86].

The dynamic and high-dimensional nature of embeddings also poses significant obstacles. Managing evolving datasets—where vectors are continuously added, modified, or deleted—calls for highly efficient incremental indexing and adaptive data maintenance. Techniques such as incremental index rebuilding [58], [87] offer potential, but further innovation is essential to support real-time updates while adhering to tight latency constraints. Additionally, effectively handling multimodal data, including

structured and unstructured information, remains a considerable computational challenge. Although hybrid frameworks like NHQ [41] provide a foundation for unified data processing, scaling such systems to accommodate diverse vector modalities—such as time-series data or hierarchical graphs—remains an open research avenue.

Looking toward the future, integrating VDBMS with revolutionary technologies offers vital opportunities to enhance their capabilities. Quantum computing, for instance, holds theoretical promise for accelerating similarity search through groundbreaking algorithms like Grover's search, though practical implementation is still in its infancy [3]. Similarly, advancements in hardware, including FPGAs and specialized vector accelerators [9], [37], point toward co-designing hardware and algorithms to maximize efficiency. Incorporating these innovations in conjunction with energy-efficient design principles can play a pivotal role in achieving sustainable data management solutions [9], [12].

Lastly, interdisciplinary research offers an untapped reservoir of potential. Cognitive science-inspired paradigms such as Hyperdimensional Computing (HDC) [20] introduce unique approaches for representing and querying vector data, while advances in large language models (LLMs) open up new possibilities for enriching semantic similarity methods [15]. Bridging expertise across fields like distributed computing, artificial intelligence, and neuromorphic engineering could transform VDBMS into systems that excel in scalability, reliability, and intelligent capabilities.

Overcoming these challenges and pursuing such opportunities will require coordinated efforts across disciplines, uniting theoretical insights with innovative system designs. By tackling these pressing questions, VDBMS have the potential to evolve into indispensable tools, fostering transformative progress across numerous domains and industries.

7 Conclusion

Vector database management systems (VDBMSs) have firmly established themselves as a cornerstone in modern data management, with their ability to process high-dimensional vector data proving indispensable for a wide spectrum of data-centric applications. From enabling semantic searches in multimodal datasets to supporting foundational workflows in machine learning, these systems exemplify a paradigm shift in addressing the challenges posed by unstructured and high-dimensional data. Distilling the insights from decades of research in similarity search, recent advances in embedding technologies, and the evolution of large-scale data applications, this survey underscores the transformative journey of VDBMSs while charting the contours of their potential.

At their core, VDBMSs derive their power from advancements in indexing methodologies, such as hierarchical navigable small-world (HNSW) graphs and locality-sensitive hashing (LSH), which have significantly accelerated approximate nearest neighbor search (ANNS) [4], [6]. Compression techniques, including product quantization and locally-adaptive methods, have further enhanced their scalability by optimizing memory usage [12], [21]. Despite these advancements, inherent trade-offs persist—indexing mechanisms optimized for precision often struggle with updates in dynamic workloads, as highlighted by the limitations of static indexing frameworks [83], [88]. Real-time index maintenance and hybrid query processing remain critical challenges, especially in heterogeneous data environments.

The ability of vector databases to unify structured and unstructured data processing within hybrid queries is a pivotal development, reflecting their growing role in multimodal integration [14]. While innovations such as learned index structures [8], [26] and hardware-accelerated designs [9] promise enhanced efficiency, vector indexing is still constrained by the curse of dimensionality and the need for robust domain-specific adaptations. Furthermore, the computational costs of large-scale deployments—accentuated by the complexity of high-dimensional vector comparisons—demand breakthroughs in energy-efficient algorithms and distributed architectures [10].

Emerging trends point toward integrating VDBMSs with advanced AI systems, like large language models (LLMs), to address limitations such as hallucinations and outdated knowledge in generative workflows [15]. Retrieval-Augmented Generation (RAG) applications exemplify how vector databases can serve as external memory for AI systems, enhancing knowledge retrieval and contextual reasoning [12].

Looking forward, the field must converge on standardization of benchmarking frameworks and develop universal metrics encapsulating accuracy, scalability, and efficiency. Research initiatives should emphasize privacy-preserving capabilities and address the intersection of vectors with emerging modalities like spatial or graph data [1], [89]. Collaborative innovation across academia and industry will be essential for overcoming the systemic obstacles of VDBMSs and unlocking their potential at the nexus of data science and AI. By addressing these open challenges, VDBMSs are poised to catalyze transformative advancements across domains, fostering a future where complex, high-dimensional data becomes seamlessly interpretable and actionable.

References

[1] J. J. Pan, J. Wang, and G. Li, "Survey of vector database management systems," *VLDB J.*, vol. 33, pp. 1591–1615, 2023. 1, 2, 5, 6, 7, 8, 11, 12, 16, 17, 19, 20

[2] T. Taipalus, "Vector database management systems: Fundamental concepts, use-cases, and current challenges," *Cognitive Systems Research*, vol. 85, 2023. 1, 9, 11, 12, 18

[3] A. Patel, A. Sands, C. Callison-Burch, and M. Apidianaki, "Magnitude: A fast, efficient universal vector embedding utility package," in *Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 120–126. 1, 2, 20

[4] W. Li, Y. Zhang, Y. Sun, W. Wang, W. Zhang, and X. Lin, "Approximate nearest neighbor search on high dimensional data — experiments, analyses, and improvement," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, pp. 1475–1488, 2016. 1, 2, 6, 12, 16, 20

[5] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazar'e, M. Lomeli, L. Hosseini, and H. J'egou, "The faiss library," *ArXiv*, vol. abs/2401.08281, 2024. 1, 3, 4, 6, 13, 14, 16, 18

[6] M. Iwasaki and D. Miyazaki, "Optimization of indexing based on k-nearest neighbor graph for proximity search in high-dimensional data," *ArXiv*, vol. abs/1810.07355, 2018. 1, 3, 6, 9, 20

[7] C. Fu, C. Wang, and D. Cai, "High dimensional similarity search with satellite system graph: Efficiency, scalability, and unindexed query compatibility," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, pp. 4139–4150, 2021. 1, 3, 6, 14, 16

[8] T. Kraska, A. Beutel, E. H. Chi, J. Dean, and N. Polyzotis, "The case for learned index structures," *Proceedings of the 2018 International Conference on Management of Data*, 2017. 1, 3, 6, 9, 13, 14, 20

[9] W. Jiang, S. Li, Y. Zhu, J. D. F. Licht, Z. He, R. Shi, C. Renggli, S. Zhang, T. Rekatsinas, T. Hoefler, and G. Alonso, "Co-design hardware and algorithm for vector search," *SC23: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16, 2023. 1, 6, 8, 11, 12, 13, 17, 19, 20

[10] R. Guo, X.-D. Luan, L. Xiang, X. Yan, X. Yi, J. Luo, Q. Cheng, W. Xu, J. Luo, F. Liu, Z. Cao, Y. Qiao, T. Wang, B. Tang, and C. Xie, "Manu: A cloud native vector database management system," *ArXiv*, vol. abs/2206.13843, 2022. 1, 4, 5, 6, 7, 8, 9, 11, 12, 13, 15, 16, 19, 20

[11] Y. Bachrach, Y. Finkelstein, R. Gilad-Bachrach, L. Katzir, N. Koenigstein, N. Nice, and U. Paquet, "Speeding up the xbox recommender system using a euclidean transformation for inner-product spaces," in *ACM Conference on Recommender Systems*, 2014, pp. 257–264. 1, 6

[12] C. Aguerrebere, M. Hildebrand, I. Bhati, T. Willke, and M. Tepper, "Locally-adaptive quantization for streaming vector search," *ArXiv*, vol. abs/2402.02044, 2024. 1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 16, 17, 18, 19, 20

[13] Q. Chen, B. Zhao, H. Wang, M. Li, C. Liu, Z. Li, M. Yang, and J. Wang, "Spann: Highly-efficient billion-scale approximate nearest neighbor search," *ArXiv*, vol. abs/2111.08566, 2021. 1, 3, 4, 5, 8, 9, 11, 14, 15, 16, 18

[14] J. Mohoney, A. Pacaci, S. R. Chowdhury, A. Mousavi, I. F. Ilyas, U. F. Minhas, J. Pound, and T. Rekatsinas, "High-throughput vector similarity search in knowledge graphs," *Proceedings of the ACM on Management of Data*, vol. 1, pp. 1 – 25, 2023. 1, 4, 7, 8, 9, 10, 11, 13, 15, 16, 18, 20

[15] Z. Jing, Y. Su, Y. Han, B. Yuan, H. Xu, C. Liu, K. Chen, and M. Zhang, "When large language models meet vector databases: A survey," *ArXiv*, vol. abs/2402.01763, 2024. 1, 2, 4, 8, 9, 11, 20

[16] H. Yang, J. Guo, J. Qi, J. Xie, S. Zhang, S. Yang, N. Li, and M. Xu, "A method for parsing and vectorization of semi-structured data used in retrieval augmented generation," *ArXiv*, vol. abs/2405.03989, 2024. 1

[17] H. Shatkay and S. Zdonik, "Approximate queries and representations for large data sequences," *Proceedings of the Twelfth International Conference on Data Engineering*, pp. 536–545, 1996. 1, 6

[18] S. Mittal, A. Joshi, and T. W. Finin, "Thinking, fast and slow: Combining vector spaces and knowledge graphs," *ArXiv*, vol. abs/1708.03310, 2017. 2, 6, 9, 13

[19] D. Kleyko, D. Rachkovskij, E. Osipov, and A. Rahimi, "A survey on hyperdimensional computing aka vector symbolic architectures, part i: Models and data transformations," *ACM Computing Surveys*, vol. 55, pp. 1 – 40, 2021. 2, 13, 18

[20] D. Kleyko, D. Rachkovskij, E. Osipov, and A. Rahim, "A survey on hyperdimensional computing aka vector symbolic architectures, part ii: Applications, cognitive models, and challenges," *ACM Computing Surveys*, vol. 55, pp. 1 – 52, 2021. 2, 20

[21] C. Aguerrebere, I. Bhati, M. Hildebrand, M. Tepper, and T. Willke, "Similarity search in the blink of an eye with compressed indices," *ArXiv*, vol. abs/2304.04759, 2023. 2, 3, 5, 6, 8, 9, 11, 12, 13, 15, 17, 18, 20

[22] A. Iscen, T. Furon, V. Gripon, M. Rabbat, and H. J. Egou, "Ieee transactions on pattern analysis and machine intelligence 1 memory vectors for similarity search in high-dimensional spaces." 2, 6, 7, 8, 11

[23] A. Gandhi, Y. Asada, V. Fu, A. Gemawat, L. Zhang, R. Sen, C. Curino, J. Camacho-Rodr'iguez, and M. Interlandi, "The tensor data platform: Towards an ai-centric database system," *ArXiv*, vol. abs/2211.02753, 2022. 2, 10

[24] J. Wang, J. Wang, G. Zeng, R. Gan, S. Li, and B. Guo, "Fast neighborhood graph search using cartesian concatenation," *2013 IEEE International Conference on Computer Vision*, pp. 2128–2135, 2013. 2, 6, 9, 13

[25] G. Tolias and O. Chum, "Asymmetric feature maps with application to sketch based retrieval," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6185–6193, 2017. 3

[26] Y. Tian, T.-K. Yan, X. Zhao, K. Huang, and X. Zhou, "A learned index for exact similarity search in metric spaces," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, pp. 7624–7638, 2022. 3, 5, 6, 9, 14, 20

[27] M. Wang, W. Xu, X. Yi, S. Wu, Z. Peng, X. Ke, Y. Gao, X. Xu, R. Guo, and C. Xie, "Starling: An i/o-efficient disk-resident graph index framework for high-dimensional vector similarity search on data segment," *Proceedings of the ACM on Management of Data*, vol. 2, pp. 1 – 27, 2024. 3, 4, 7, 8, 11, 13, 15, 16, 18

[28] A. Al-Mamun, H. Wu, Q. He, J. Wang, and W. Aref, "A survey of learned indexes for the multi-dimensional space," *ArXiv*, vol. abs/2403.06456, 2024. 3, 7, 9, 10, 13

[29] G. Teodoro, E. Valle, N. Mariano, R. Torres, W. M. Jr, and J. Saltz, "Approximate similarity search for online multimedia services on distributed cpu–gpu platforms," *The VLDB Journal*, vol. 23, pp. 427 – 448, 2012. 3, 8, 11

[30] B. Lu, J. Ding, E. Lo, U. F. Minhas, and T. Wang, "Apex: A high-performance learned index on persistent memory," *Proc. VLDB Endow.*, vol. 15, pp. 597–610, 2021. 3, 12, 18

[31] J. Liu, K. Zhao, P. Sommer, S. Shang, B. Kusy, and R. Jurdak, "Bounded quadrant system: Error-bounded trajectory compression on the go," *2015 IEEE 31st International Conference on Data Engineering*, pp. 987–998, 2014. 4

[32] P. Krishnan, L. Vadlamani, and L. P. Natrajan, "Coded data rebalancing: Fundamental limits and constructions," *2020 IEEE International Symposium on Information Theory (ISIT)*, pp. 640–645, 2020. 4

[33] J. Wu, J. Xiong, H. Dai, Y. Wang, and C. Xu, "Mix-rs: A multi-indexing system based on hdfs for remote sensing data storage," *ArXiv*, vol. abs/2208.02987, 2022. 4

[34] H. Bhatia, D. Hoang, G. Morrison, W. Usher, V. Pascucci, P. Bremer, and P. Lindstrom, "Amm: Adaptive multilinear meshes," *IEEE Transactions on Visualization and Computer Graphics*, vol. PP, pp. 1–1, 2020. 4

[35] R. Chen, Y. Peng, X. Wei, H. Xie, R. Chen, S. Shen, and H. Chen, "Characterizing the dilemma of performance and index size in billion-scale vector search and breaking it with second-tier memory," *ArXiv*, vol. abs/2405.03267, 2024. 4, 8, 10, 18

[36] K. Shanmugam, D. Papailiopoulos, A. Dimakis, and G. Caire, "A repair framework for scalar mds codes," *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 1166–1173, 2012. 4

[37] M. Perotti, M. A. Cavalcante, N. Wistoff, R. Andri, L. Cavigelli, and L. Benini, "A "new ara" for vector computing: An open source highly efficient risc-v v 1.0 vector processor design," *2022 IEEE 33rd International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, pp. 43–51, 2022. 5, 20

[38] D. Kleyko, M. Davies, E. P. Frady, P. Kanerva, S. J. Kent, B. Olshausen, E. Osipov, J. Rabaey, D. Rachkovskij, A. Rahimi, and F. Sommer, "Vector symbolic architectures as a computing framework for emerging hardware," *Proceedings of the IEEE*, vol. 110, pp. 1538–1571, 2021. 5, 16

[39] T. Yang, W. Hu, W. Peng, Y. Li, J. Li, G. Wang, and X. Liu, "Vdtuner: Automated performance tuning for vector data management systems," *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pp. 4357–4369, 2024. 5, 8, 12, 15, 16

[40] M. A. Cavalcante, D. Wüthrich, M. Perotti, S. Riedel, and L. Benini, "Spatz: A compact vector processing unit for high-performance and energy-efficient shared-l1 clusters," *2022 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pp. 1–9, 2022. 5, 8, 19

[41] M. Wang, L. ang Lv, X. Xu, Y. Wang, Q. Yue, and J. Ni, "Navigable proximity graph-driven native hybrid queries with structured and unstructured constraints," *ArXiv*, vol. abs/2203.13601, 2022. 5, 7, 10, 12, 14, 20

[42] C. Marchet, L. Lecompte, A. Limasset, L. Bittner, and P. Peterlongo, "A resource-frugal probabilistic dictionary and applications in bioinformatics," *ArXiv*, vol. abs/1703.00667, 2017. 5, 14, 17

[43] Z. Li, "Geospatial big data handling with high performance computing: Current approaches and future directions," *ArXiv*, vol. abs/1907.12182, 2019. 5

[44] M. Drosou and E. Pitoura, "Disc diversity: result diversification based on dissimilarity and coverage," *Proc. VLDB Endow.*, vol. 6, pp. 13–24, 2012. 6

[45] Y. Han, C. Liu, and P. Wang, "A comprehensive survey on vector database: Storage and retrieval technique, challenge," *ArXiv*, vol. abs/2310.11703, 2023. 6, 11, 12, 15, 19

[46] F. André, A.-M. Kermarrec, and N. L. Scouarnec, "Quicker adc : Unlocking the hidden potential of product quantization with simd," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, pp. 1666–1677, 2018. 6, 7, 11, 13

[47] I. Munro, Y. Nekrich, and J. Vitter, "Dynamic data structures for document collections and graphs," *Proceedings of the 34th ACM*

SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, 2015. 6, 7

[48] Y. Xu, J. Gao, Y. Gou, C. Long, and C. S. Jensen, "irangegraph: Improvising range-dedicated graphs for range-filtering nearest neighbor search," *ArXiv*, vol. abs/2409.02571, 2024. 7

[49] J. J. Lin, R. Pradeep, T. Teofili, and J. Xian, "Vector search with openai embeddings: Lucene is all you need," *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 2023. 7, 14

[50] A. Singh, S. J. Subramanya, R. Krishnaswamy, and H. Simhadri, "Freshdiskann: A fast and accurate graph-based ann index for streaming similarity search," *ArXiv*, vol. abs/2105.09613, 2021. 7, 9, 10, 13, 14, 16, 18

[51] T. Gagie, J. I. González-Nova, S. Ladra, G. Navarro, and D. Seco, "Faster compressed quadtrees," *2015 Data Compression Conference*, pp. 93–102, 2014. 7

[52] D. Lemire and L. Boytsov, "Decoding billions of integers per second through vectorization," *Software: Practice and Experience*, vol. 45, pp. 1 – 29, 2012. 8

[53] M. Tang, Y. Yu, W. Aref, A. R. Mahmood, Q. Malluhi, and M. Ouzzani, "Locationspark: In-memory distributed spatial query processing and optimization," *Frontiers in Big Data*, vol. 3, 2019. 8

[54] S. Jaiswal, R. Krishnaswamy, A. Garg, H. Simhadri, and S. Agrawal, "Ood-diskann: Efficient and scalable graph anns for out-of-distribution queries," *ArXiv*, vol. abs/2211.12850, 2022. 9, 13, 15, 17, 18

[55] T. Bingmann, P. Bradley, F. Gauger, and Z. Iqbal, "Cobs: a compact bit-sliced signature index," in *SPIRE*, 2019, pp. 285–303. 10

[56] A. Hadian, A. Kumar, and T. Heinis, "Hands-off model integration in spatial index structures," *ArXiv*, vol. abs/2006.16411, 2020. 10

[57] H. Lan, Z. Bao, J. Culpepper, and R. Borovica-Gajic, "Updatable learned indexes meet disk-resident dbms - from evaluations to design choices," *Proceedings of the ACM on Management of Data*, vol. 1, pp. 1 – 22, 2023. 10

[58] Z. Bao, L.-L. Liu, Z. Wu, Y. Zhou, D. Fan, M. Aibin, Y. Coady, and A. Brownsword, "Delta tensor: Efficient vector and tensor storage in delta lake," *ArXiv*, vol. abs/2405.03708, 2024. 11, 19

[59] W. Wu, J. He, Y. Qiao, G. Fu, L. Liu, and J. Yu, "Hqann: Efficient and robust similarity search for hybrid queries with structured and unstructured constraints," *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022. 11

[60] J. Rygl, J. Pomikálek, R. Rehurek, M. Růžička, V. Novotný, and P. Sojka, "Semantic vector encoding and similarity search using fulltext search engines," in *Rep4NLP@ACL*, 2017, pp. 81–90. 12

[61] J. Li, H.-F. Liu, C. Gui, J. Chen, Z. Ni, N. Wang, and Y. Chen, "The design and implementation of a real time visual search system on jd e-commerce platform," *Proceedings of the 19th International Middleware Conference Industry*, 2018. 12, 13

[62] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. Summers, "Chestx-ray: Hospital-scale chest x-ray database and benchmarks on weakly supervised classification and localization of common thorax diseases," in *Deep Learning and Convolutional Neural Networks for Medical Imaging and Clinical Informatics*, 2019, pp. 369–392. 13, 15

[63] W. Wang, M. Zhang, G. Chen, H. V. Jagadish, B. Ooi, and K. Tan, "Database meets deep learning: Challenges and opportunities," *ArXiv*, vol. abs/1906.08986, 2016. 13

[64] J. Wu, Y. Zhang, S. Chen, J. Wang, Y. Chen, and C. Xing, "Updatable learned index with precise positions," *Proc. VLDB Endow.*, vol. 14, pp. 1276–1288, 2021. 14

[65] J. Ding, U. F. Minhas, H. Zhang, Y. Li, C. Wang, B. Chandramouli, J. Gehrke, D. Kossmann, and D. Lomet, "Alex: An updatable adaptive learned index," *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, 2019. 14, 18

[66] R. Chikhi, J. Holub, and P. Medvedev, "Data structures to represent a set of k-long dna sequences," *ACM Computing Surveys (CSUR)*, vol. 54, pp. 1 – 22, 2021. 14

[67] L. Chen, Y. Gao, X. Song, Z. Li, X. Miao, and C. S. Jensen, "Indexing metric spaces for exact similarity search," *ACM Computing Surveys*, vol. 55, pp. 1 – 39, 2020. 15

[68] M. Gagolewski, "Data fusion: Theory, methods, and applications," *ArXiv*, vol. abs/2208.01644, 2022. 15

[69] S. K. Jensen, T. Pedersen, and C. Thomsen, "Scalable model-based management of correlated dimensional time series in modelardb+," *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pp. 1380–1391, 2019. 15

[70] V. Enes, P. S. Almeida, C. Baquero, and J. Leitao, "Efficient synchronization of state-based crdts," *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pp. 148–159, 2018. 15

[71] N. Stephens, S. Biles, M. Boettcher, J. Eapen, M. Eyole, G. Gabrielli, M. Horsnell, G. Magklis, A. Martínez, N. Prémillieu, A. Reid, A. Rico, and P. Walker, "The arm scalable vector extension," *IEEE Micro*, vol. 37, pp. 26–39, 2017. 15

[72] H. Abu-Libdeh, D. Altinbuken, A. Beutel, E. H. Chi, L. Doshi, T. Kraska, X. Li, A. Ly, and C. Olston, "Learned indexes for a google-scale disk-based database," *ArXiv*, vol. abs/2012.12501, 2020. 15

[73] B. Peng, P. Fatourou, and T. Palpanas, "Paris+: Data series indexing on multi-core architectures," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, pp. 2151–2164, 2020. 16

[74] M. Linardi and T. Palpanas, "Scalable data series subsequence matching with ulisse," *The VLDB Journal*, vol. 29, pp. 1449 – 1474, 2020. 16

[75] J. Arulraj, A. Pavlo, and K. T. Malladi, "Multi-tier buffer management and storage system design for non-volatile memory," *ArXiv*, vol. abs/1901.10938, 2019. 16

[76] F. Kurpicz, "Engineering compact data structures for rank and select queries on bit vectors," *ArXiv*, vol. abs/2206.01149, 2022. 17

[77] U. Bauer, X. Ge, and Y. Wang, "Measuring distance between reeb graphs," *Proceedings of the thirtieth annual symposium on Computational geometry*, 2013. 17

[78] F. Basık, H. Ferhatosmanoğlu, and B. Gedik, "Slim: Scalable linkage of mobility data," *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, 2020. 17

[79] T. Friedman and G. V. den Broeck, "Symbolic querying of vector spaces: Probabilistic databases meets relational embeddings," *ArXiv*, vol. abs/2002.10029, 2020. 17

[80] D. Tomaszuk and D. Hyland-Wood, "Rdf 1.1: Knowledge representation and data integration language for the web," *ArXiv*, vol. abs/2001.00432, 2020. 17

[81] S. Arya, G. D. D. Fonseca, and D. Mount, "Optimal approximate polytope membership," *ArXiv*, vol. abs/1612.01696, 2016. 18

[82] Z. Wang, J. Huang, Z. Sun, D. Cohen-Or, and M. Lu, "Layered image vectorization via semantic simplification," *ArXiv*, vol. abs/2406.05404, 2024. 18

[83] H. Wang, "Unit-disk range searching and applications," in *Scandinavian Workshop on Algorithm Theory*, 2022, pp. 32:1–32:17. 19, 20

[84] K. Saur, T. Dumitras, and M. Hicks, "Evolving nosql databases without downtime," *2016 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp. 166–176, 2015. 19

[85] A. Dinh, J. Wang, S. Wang, G. Chen, W. Chin, Q. Lin, B. Ooi, P. Ruan, K. Tan, Z. Xie, H. Zhang, and M. Zhang, "Ustore: A distributed storage with rich semantics," *ArXiv*, vol. abs/1702.02799, 2017. 19

[86] A. Filtser, O. Filtser, and M. J. Katz, "Approximate nearest neighbor for curves: Simple, efficient, and deterministic," *Algorithmica*, vol. 85, pp. 1490–1519, 2019. 19

[87] H. Kaplan, W. Mulzer, L. Roditty, P. Seiferth, and M. Sharir, "Dynamic planar voronoi diagrams for general distance functions and their algorithmic applications," *Discrete & Computational Geometry*, vol. 64, pp. 838 – 904, 2016. 19

[88] D. Tian and D. Tao, "Global hashing system for fast image search," *IEEE Transactions on Image Processing*, vol. 26, pp. 79–89, 2019. 20

[89] M. Besta, E. Peter, R. Gerstenberger, M. Fischer, M. Podstawski, C. Barthels, G. Alonso, and T. Hoefler, "Demystifying graph databases: Analysis and taxonomy of data organization, system designs, and graph queries," *ACM Computing Surveys*, vol. 56, pp. 1 – 40, 2019. 20