arXiv:2601.01937v1 [cs.DB] 5 Jan 2026

面向未来的向量搜索：从内存驻留、静态异构存储到云原生架构

Yitong Song 香港浸会大学 yitong_song@hkbu.edu.hk

Xuanhe Zhou 上海交通大学 zhouxuanhe@sjtu.edu.cn

Christian S. Jensen 奥尔堡大学 丹麦 csj@cs.aau.dk

Jianliang Xu 香港浸会大学 xujl@comp.hkbu.edu.hk

摘要

向量搜索（VS）已成为多模态数据管理的基础组成部分，支持图像、视频和代码检索等核心功能。随着向量数据迅速增长，向量搜索在平衡搜索性能、延迟、可扩展性和成本方面面临着日益严峻的挑战。向量搜索的发展与存储架构的变化密切相关。早期的向量搜索方法依赖全内存设计以实现低延迟，但其可扩展性受内存容量和成本限制。为此，近期研究采用异构架构，将空间密集型向量和索引结构卸载到固态硬盘，同时利用块局部性和 I/O 高效策略，在十亿规模下保持较高的搜索性能。展望未来，对万亿规模向量检索和云原生弹性的需求日益增长，正推动架构进一步转向内存—固态硬盘—对象存储，以实现经济高效的数据分层和无缝扩展。

在本教程中，我们从存储架构的视角回顾向量搜索技术的发展历程。首先，我们回顾内存驻留方法，涵盖经典的基于倒排文件、哈希、量化和图的设计。随后，我们系统概述异构存储向量搜索技术，包括其索引设计、块级布局、查询策略和更新机制。最后，我们探讨新兴的云原生系统，并指出未来大规模向量检索系统中有待研究的开放问题。

关键词

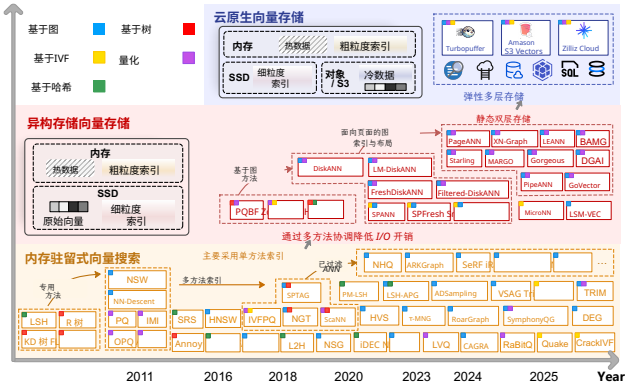
高维向量、向量搜索、异构存储

1 引言

向量搜索（VS）旨在高维嵌入空间中检索语义相似的项目，已成为大规模多模态数据搜索的基础构件，支持诸如淘宝中的图像到图像检索 [53],TikTok 中的相似视频检索 [42], 以及 Trae [11] 等代码助手等应用。随着应用需求从百万级增长到十亿级，如今更达到万亿级，向量数据的存储和索引结构

术的发展已从纯粹的内存设计演进至异构

允许为个人或课堂使用而免费制作本作品的全部或部分数字副本或纸质副本，但不得为营利或商业目的制作或分发副本，且副本首页必须附有本声明和完整引文。对于本作品中由作者以外的其他方拥有版权的组成部分，必须尊重其版权。注明出处后允许摘录。除此之外，若要复制、再版、发布到服务器或分发给他人，均须事先获得特别许可和/或支付费用。请通过 permissions@acm.org 申请许可。会议缩写“*XX*”，纽约州伍德斯托克 © 2018年版权归所有者/作者所有。出版权授权给 ACM。ACM ISBN 978-1-4503-XXXX-X/18/06 https://doi.org/XXXXXXX.XXXXXXX



Fi图 1：向量搜索技术的评估方法。

内存–SSD部署，并进一步迈向云原生架构。存储架构的这一转变推动了VS技术的显著演进（见图1）。

(1) 内存驻留式向量搜索。早期的向量搜索方法将原始向量和索引结构都完全存储在内存中，通过优化计算、基本忽略数据访问成本，实现极高的性能。在这种设置下，研究者开发了多种技术，包括基于树、 [5, 40], 基于倒排文件 [1, 36, 39, 54], [14, 18, 25, 48, 49], [2 4, 10, 12, 基 16, 21], [9, 29, 35, 38, 44, 61] 于哈希、基于量化和基于图的方法。然而，随着向量规模不断增长，而内存成本依然高昂，将向量和内存密集型索引都完全存储在内存中已不切实际。***(2)*** 静态异构存储向量搜索。为解决内存驻留式向量搜索在可扩展性和成本方面的瓶颈，人们开发了多种异构存储向量搜索方法。此类方法始于 PQBF2017 [33] ，并自2024年以来迅速发展，出现了 Starling [50], 、PipeANN [15], 和 PageANN [22], 等方法。大多数方法采用静态双层内存–SSD架构。在这种设计中，占用空间较大的原始向量和细粒度索引结构存放在固态硬盘上，而压缩向量或粗粒度导航结构则保留在内存中。在此架构基础上，近期方法进一步利用SSD友好布局、块级局部性和I/O高效搜索策略，以降低数据访问开销并实现低延迟性能。

（***3***）弹性多层级向量搜索。随着向量数据接近万亿规模，静态的内存—固态硬盘架构面临成本性能权衡，且缺乏弹性。这推动了向弹性多层设计的转变（尤其是在云原生场景中 [17,28]），将内存、固态硬盘和对象/S3存储整合到统一且开销低的架构中。Zilliz Cloud [17], 、Amazon S3 Vectors [58], 和 Turbopuffer [19] 等现代系统采用

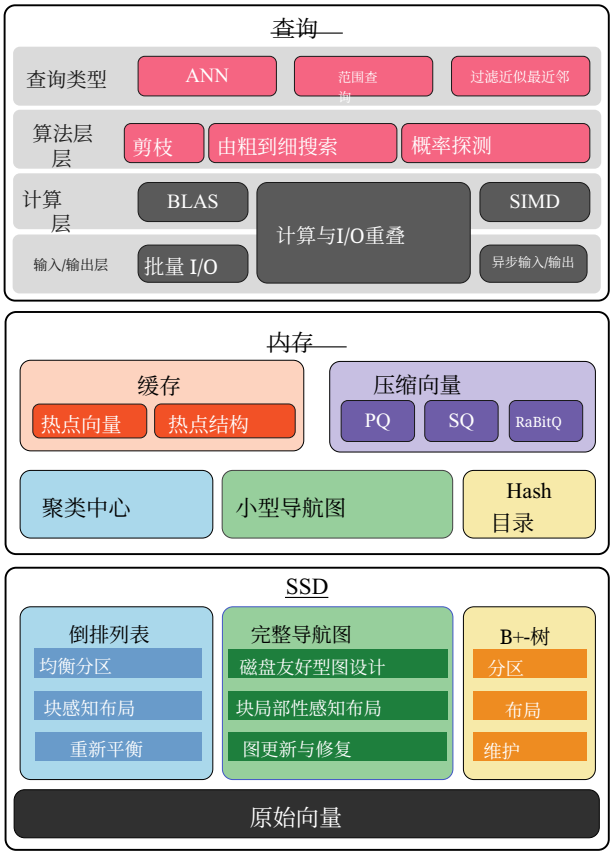


图2：异构存储向量搜索技术概览。

这种模式，根据访问模式和成本性能考量，将向量和索引组件动态部署到不同层级。对象存储几乎能够提供无限容量，并且可独立于计算资源进行扩展，从而实现无缝扩容；同时，按需将频繁访问的数据提升至更热的层级（例如固态硬盘或内存）。这种弹性且能够感知工作负载的迁移策略，使多层级向量搜索非常适用于万亿级向量搜索场景。

受这些进展的启发，本教程全面概述了在三种存储架构下发展起来的向量搜索技术。我们首先简要回顾内存驻留式方法，随后深入探讨迅速扩展的异构存储向量搜索技术。最后，我们重点介绍新兴的云原生实践，并概述弹性多层级架构带来的未来研究机会。**教程概述**。本次为期3小时的教程根据存储架构分为三个部分。

第1部分（0.5小时）：内存驻留式向量搜索。我们简要回顾经典的内存向量搜索方法，并介绍具有代表性的基于倒排文件、基于哈希、基于量化和基于图的技术在完全内存驻留的环境下如何运作，为后续章节奠定技术基础。

第 2 部分（2 小时）：异构存储向量搜索。这是本教程的主要部分。我们将介绍一个统一技术架构（图 2），它概括了现代内存-固态硬盘方法中的常见设计模式。为清晰、有条理地阐述内容，我们将现有研究分为倒排文件、图和基于树的类别，并讨论各类别中的技术。

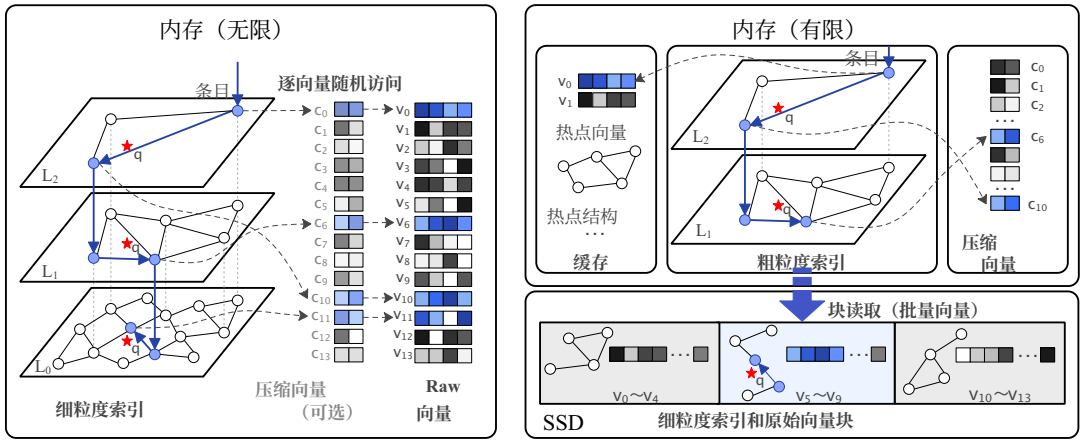
(1) 基于 *IVF* 的方法（*40* 分钟）。这些方法将向量划分为多个簇，并将查询处理限制在仅探测少量相关簇的范围内。我们将讨论四个关键组成部分：(a) 聚类与分区，(b) 块感知倒排列表布局，(c) 基于探测的查询执行与优化，以及 (d) 更新与维护机制。

(2) 基于图的方法（*60* 分钟）。这些方法将向量组织成邻近图，并通过遍历这些图来回答查询。我们将从以下四个方面介绍这些技术：(a) 磁盘友好的图设计，(b) 块局部性感知布局，(c) 由粗到细的搜索与优化，以及 (d) 图更新与修复。(3) 基于树的方法（*20* 分钟）。这些方法借助 B+-树等目录结构，将向量组织到磁盘块中，并通常结合哈希函数或学习型模型，将高维向量映射为一维键。我们将重点介绍三个方面：(a) 面向块的划分与布局，(b) 块过滤与查询执行，以及 (c) 更新机制。

第 3 部分（0.5 小时）：弹性多层向量搜索。我们重点介绍两种新兴的向量搜索系统 Ziliz Cloud [17]和 Turbopuffer [19]，它们采用多层级存储，并探讨未来的研究方向。**对数据管理领域的重要性**。随着数据库越来越多地纳入向量数据以支持基于嵌入的搜索，向量搜索正迅速成为现代数据库管理系统中的一等工作负载。然而，数据管理领域仍缺乏全面的认识，尚不清楚如何在大规模场景下高效存储、索引和更新向量数据。本教程将阐明以下内容，为向量搜索的开发提供实用指导和基本设计原则：（1）不同存储层级如何重塑向量搜索方法的设计；（2）如何设计内存-SSD 向量存储方法以应对主要的 I/O 瓶颈；（3）云原生向量存储系统如何从分层存储架构中受益，以及面临哪些挑战。

目标受众与学习成果。本教程面向 SIGMOD 与会者，包括对向量检索、向量数据库或多模态检索系统感兴趣的学术界和业界人士。我们将从内存型向量存储技术讲起，通过直观的解释和启发性的示例，从头建立基础概念，随后逐步扩展到内存-SSD 向量存储和多层级存储架构。教程内容自成体系，无需任何先备知识。与会者将了解存储架构如何影响向量搜索方法，并掌握构建可扩展且高效的向量存储系统的实用原则。

与现有教程的区别。现有教程主要关注内存型向量存储算法 [7] 或实用的向量数据库管理系统[43]。然而，自这些教程发布以来，一系列研究 [15, 17, 19, 22, 24,34, 50, 52, 56, 57, 59]推动向量搜索超越了内存范式，引入了异构存储设计和弹性多层级系统，同时也带来了新的瓶颈与设计原则，而这些并非早期教程的关注重点。本教程针对这些变化，从以存储架构为中心的视角出发，系统梳理向量搜索技术从内存驻留设计，到异构内存-SSD 解决方案，再到云原生多层级架构的发展历程。我们深入探讨内存-SSD 方法如何应对主要的 I/O 瓶颈，以及弹性多层向量搜索的架构设计。



（a）内存驻留式向量搜索　　（b）内存-SSD 向量搜索

图 3：内存驻留式向量搜索与内存-SSD 向量搜索的对比（以基于图的索引为例）。

2 教程大纲

本节概述教程的内容与结构，沿着向量搜索在三种存储架构中的演进展开：内存驻留、异构存储和弹性多层设计。

2.1 内存驻留式向量搜索

内存驻留式向量搜索方法将原始向量和索引结构全部存储在内存中，从而实现低延迟数据访问和快速查询执行。我们将介绍四种具有代表性的内存方法：基于 IVF、基于哈希、基于量化和基于图的方法。

基于 IVF 的方法。倒排文件（IVF）技术使用诸如 *k*-means [23] 之类的算法将向量划分为若干簇，并将每个向量分配给最近的质心。每个簇都对应一个倒排列表，其中存储原始向量或其压缩表示。查询时，查询向量首先通过与质心进行比较来确定*nprobe* 最相关的簇，然后仅对这些簇中的向量进行进一步评估。IVF 实现简单，也易于支持更新。不过，它只能进行粗粒度剪枝，因为被探测的簇中可能仍有许多向量需要扫描。

基于哈希的方法。 基于哈希的方法，尤其是局部敏感哈希（LSH），利用专门设计的哈希函数将向量映射到各个桶中，使相似向量很可能落入同一个桶。查询时，查询向量以相同方式进行哈希，从而将搜索范围限制在哈希码相似或相同的桶中。这种方法为向量存储提供了理论保证：返回结果经证明在有界误差范围内接近真实结果，而且查询时间在渐近意义上可以低于线性扫描，因此在理论上颇具吸引力。然而，在实践中，基于哈希的方法难以实现高准确率，因为这通常需要较长的哈希码和多个哈希表，进而导致桶稀疏、内存占用过高以及性能下降。

基于量化的方法。 这些方法将原始向量压缩为紧凑编码，使得仅凭这些编码就能以低得多的计算成本估算查询向量与数据向量之间的距离。其中，乘积量化（PQ） [21] 是应用最广泛的技术之一：它将每个向量拆分为多个低维子空间，并在每个子空间中通过聚类学习一个码本；随后，每个子向量都以其最近质心的索引进行编码。查询时，仅使用紧凑编码即可获得近似距离，从而避免对原始向量执行开销高昂的运算。这种方法具有出色的内存效率，且距离计算速度快。然而，压缩不可避免地会引入近似误差，从而可能降低查询准确率。

基于图的方法。 这类方法将每个向量与其在不同方向上的近似最近邻相连，构建支持高效贪心导航的邻近图。查询从一个或多个入口点开始，迭代探索能够使其更接近目标向量的邻居；当没有候选向量能够带来进一步改进时，搜索便会停止。HNSW [38] 等代表性结构还通过将向量组织成分层图，进一步提升查询效率：上层包含逐渐变稀疏的图，引导搜索快速到达真实最近邻附近的区域；下层则提供细粒度检索，以便进一步优化结果。这种分层导航大幅缩短了搜索路径并降低了查询延迟。基于图的方法通常在内存型向量存储技术中实现最佳的准确率与效率权衡，并主导着许多高性能应用 [27, 51]。然而，这类图的内存占用很高，而且搜索过程涉及大量随机数据访问，因此难以将其扩展到驻磁盘环境。

2.2 异构存储向量搜索

为了支持十亿级部署并保持低延迟搜索，近期方法采用异构内存-SSD存储架构：内存存储用于路由和粗粒度剪枝的轻量级结构，而固态硬盘则存放占用空间较大的原始向量和细粒度索引。与内存驻留式向量搜索不同，后者可以依赖单一索引范式并专注于降低计算成本；异构存储方法通常会结合多个互补索引（例如，量化 + 图、哈希 + 树），并带来如何将随机输入/输出降至最低的新挑战。

表 1：代表性内存-SSD VS方法的比较。

Type	方法	关键技术	存储		查询类型			Up.-Opt.	应用
			DRAM	SSD	近似最近邻最近邻搜索	范围搜索	过滤近似		
IVF-基于	SPANN [6]	分层平衡聚类	质心	倒排列表 + 向量	✓	×	×	×	动态工作负载
	SPFresh [55]	轻量级再平衡	质心	倒排列表 + 向量	✓	×	×	✓	
	MicroNN [45]	设备端轻量级聚类重组	质心	倒排列表 + 向量	✓	×	×	✓	
	Zoom [60]	内存近似搜索与 SSD 精确重新排序	紧凑向量 + 质心 + 倒排列表	向量	✓	×	×	×	
图-式	DiskANN [20]	Vamana 图	紧凑向量	Vamana +向量	✓	×	×	×	延迟关键，高精度搜索
	带过滤的-DiskANN [13]	带过滤的 Vamana 支持	紧凑向量	Vamana +向量	✓	×	✓	×	
	新鲜度-DiskANN [46]	动态更新策略	紧凑向量	Vamana +向量	✓	×	×	✓	
	Starling [50]	优化布局	紧凑向量 + 条目图	Vamana +向量	✓	✓	×	×	
	LSM-VEC [62]	HNSW + 日志结构合并树	上层图	底层图	✓	×	×	✓	
	PageANN [22]	页面感知布局	紧凑向量 + 条目图	图 +向量	✓	×	×	×	
基于树的基础	I-LSH[31] / EL-LSH [32]	查询感知LSH	哈希表	多重 B^+-树	✓	×	×	×	低至中等维度
	NeOPFA [26]	学习型有序列表与顺序页面访问	哈希表	有序列表（索引由 B^+-树） +向量	✓	×	×	×	
	PQBF [33]	基于PQ的过滤，结合树剪枝	PQ 元数据 + 目录	紧凑向量+ PQB^+-树	✓	×	×	×	

* RS、FANN 和优化更新支持表示范围 搜索、过滤近似最近邻搜索（带属性谓词的近似最近邻），以及该方法是否提供优化的更新支持，分别 ly.

在本教程中，我们根据核心精细筛选结构对现有异构内存-SSD解决方案进行分类，即基于IVF、基于图和基于树的方法，并介绍相应的设计原则与优化策略。表 1 总结了每个类别中的代表性方法，以及它们所面向的查询类型和应用场景。

2.2.1 基于 *IVF* 的方法。 异构内存–固态硬盘存储上的基于 IVF 的向量搜索方法，通过将聚类中心和轻量级路由结构保存在内存中，同时将原始（或压缩后的）向量及完整倒排列表存储在固态硬盘上，扩展了经典的内存中 IVF 范式。其关键挑战在于设计平衡聚类和固态硬盘感知布局，使查询探测能够通过少量保持局部性的块读取完成，而不是执行代价高昂且分散的随机输入/输出。我们通过四个关键组件介绍相关技术：（i）聚类与分区，将向量分配到平衡的簇中，以维持有界的列表大小并增强共同访问局部性；（ii）块感知倒排列表布局，对列表片段进行对齐和分组，以提高页面效率并减少跨块获取；（iii）基于探测的查询执行与优化，控制被探测簇的数量，并采用进一步降低延迟的技术；以及（iv）更新与维护机制，在频繁更新的情况下保持簇平衡和布局局部性。

聚类与分区。大多数方法采用平衡 *k*-均值 [30, 37] 来控制列表大小，或采用小批量 *k*-均值 [1, 45] 在内存有限的情况下运行。在固态硬盘驻留场景中，平衡聚类尤为重要，因为它有助于使倒排列表与SSD块对齐，并避免产生过大的簇，以免探测时造成过多页面读取。SPANN [6] 还采用层次平衡聚类，以由粗到细的方式优化路由，从而减少探测的簇数量并提高数据局部性。

块感知倒排列表布局。SSD驻留型倒排文件（IVF）的一项关键挑战是组织倒排列表，以尽量减少输入/输出。常见策略包括块对齐列表段 [1],、增强局部性的连续布局（将相似的簇分组） [6, 45], ，以及将倒排列表划分为冷热区域 [28]。对于过滤近似最近邻搜索（FANN）工作负载，Hybrid-IVFFlat [8] 会在每个倒排列表条目中将向量标识符与可过滤属性放在一起，从而无需额外的随机读取，即可通过一次顺序扫描完成过滤和候选剪枝。此外，它的倒排列表还会进一步划分为可独立加载的段，因此只需读取与过滤谓词相关的段。

基于探测的查询执行与优化。查询期间，系统会获取质心最接近查询的前*nprobe* 个簇。选择 *nprobe* 至关重要：取值过小会损害召回率，取值过大则会增加输入/输出成本。为降低延迟，近期方法采用自适应列表选择、批量 *I/O*、预取和计算与*I/O*重叠，以提高簇访问效率。Zoom [60] 采用预览–重排序策略，先通过紧凑向量预览筛选候选项，再获取驻留于固态硬盘上的向量，从而减少输入/输出量和尾延迟。SPANN [6] 使用层次聚类，将探测范围限制在局部化分区内，从而减少每次查询访问的列表数量。

更新与维护机制。 基于IVF的方法天然支持动态更新。插入操作只需将向量追加到相应的倒排列表中即可处理，而删除操作只需对列表进行轻量修改。然而，频繁更新可能导致簇大小失衡并降低查询效率。为此，MicroNN [45] 引入了一种增量存储，用于临时缓冲新增或更新的向量，并定期在后台将其合并到主倒排列表中

，从而避免立即干扰固态硬盘驻留数据。SPFresh [55] 专注于维持簇平衡，通过轻量级再平衡有选择地在簇之间迁移少量向量，在无需进行成本高昂的完整重新聚类的情况下保持分区质量。

2.2.2 基于图的方法。 基于图的向量搜索方法将内存中的图索引（例如 HNSW [38], Vamana [20]）扩展到内存与固态硬盘异构架构：在内存中保留轻量级导航结构和紧凑向量，同时将完整的邻接列表和原始向量存储在固态硬盘上。这种转变带来了一个根本性的新挑战：每次邻居扩展都可能触发页面获取，因此必须重新设计图，使遍历能够以块大小的批次进行，并具有良好的局部性。与数据访问几乎无需成本的内存图不同，驻留于固态硬盘的图必须谨慎调整图结构和物理布局，在不牺牲准确率的前提下尽量减少输入/输出。我们将现有技术归纳为四个核心组成部分：（i）磁盘友好型图设计，通过限制度数并确保方向多样性，使邻接列表能够与固态硬盘块对齐；（ii）块局部性感知布局，将邻居和原始向量组织到保持局部性的固态硬盘块中；（iii）由粗到细搜索与优化，采用批量I/O、预取和计算与I/O重叠来降低延迟；（iv）图更新与修复策略，在动态工作负载下维持图质量和块局部性。

磁盘友好型图设计。 与内存驻留图不同，SSD驻留图必须维护规模小、保留局部性且按块对齐的邻接列表，以减少随机输入/输出。Vamana [20]、XN-Graph [59]、BAMG [24],和 PageANN [22] 等代表性设计通过控制度数和边的多样化，使图既便于导航又对SSD友好。Vamana采用基于遮挡的剪枝，并结合进度参数保留长程边，确保度数固定，以适应页面对齐。XN-Graph在应用基于遮挡的剪枝前，通过扩展邻域下降扩大邻域覆盖范围，从而增强长程连通性，提升在SSD上的可导航性。BAMG区分块内链接和跨块链接，优先选择能够推动搜索进展且无需额外页面读取的边。PageANN则进一步将向量聚合为页面级节点，并移除冗余的页内链接，以匹配物理页面布局。

块局部性感知布局。SSD驻留图面临的一项关键设计挑战是，如何将邻接列表、原始向量和元数据放置到SSD块中，以尽量减少输入/输出。现有方法大致分为两类：耦合布局和解耦布局。耦合布局 [20, 22, 50, 57] 将向量及其邻居放在同一页面中，因此一次读取即可同时支持向量评估和邻居扩展。解耦布局 [24, 34, 47] 将图结构与原始向量分开存储，使两者能够分别针对各自的访问模式进行优化：图索引组件会被频繁访问，而完整向量仅在最终重新排序时才需要。然而，基于插入顺序或ID顺序的朴素邻居列表布局无法保证图邻近的邻居位于同一页面，导致遍历过程中频繁跳转页面。为解决这一问题，近期研究将块布局建模为局部性优化问题，将彼此可达的邻居紧密地分组到相同页面中。DiskANN++ [41] 会重新组织节点采用星形装箱/分箱方式，使图邻近顶点位于按页面对齐的单元中；Starling [50] 和MARGO [57] 最大化页内邻居重叠，使每次读取都能获得更多可用邻居；PageANN [22] 则进一步完善了这一思路，通过将相似向量聚类成页面级“超级节点”，使逻辑图导航与物理SSD页精确对齐，并消除冗余的页内边。

由粗到细搜索与优化。 SSD驻留图搜索仍遵循最佳优先范式，但与邻居可即时访问的内存中遍历不同，每次扩展节点都可能产生高昂的SSD读取开销。为缓解这一问题，近期系统采用由粗到细的搜索策略：首先利用低成本的内存结构（例如入口图、紧凑向量）引导导航至有希望的区域，仅在必要时才进行细粒度SSD访问。Starling [50] 展示了这一做法：先利用小型内存入口图快速完成初始路由，再访问SSD驻留邻居；PageANN [22]则将剪枝进一步推至页级遍历阶段，仅加载其聚合向量可能有助于最终结果的页面，从而提升每次读取的效用。为进一步隐藏I/O延迟，搜索算法还结合了异步I/O、预取和计算与I/O重叠，并采用轻量级剪枝（例如束大小控制 [50], 下界过滤 [47]），以减少每个页面扩展的节点数。

图更新与修复策略。在动态工作负载下维持高性能，对驻留于固态硬盘上的图而言尤其具有挑战性，因为插入操作和删除操作会迅速降低图的可导航性和块局部性。FreshDiskANN [46] 将新向量插入一个小型内存动态图和专用的新鲜SSD区域，同时将删除操作记录为墓碑标记。系统定期通过删除–插入–修补流水线执行后台流式合并，将这些更新整合到主图中。LSM-VEC [62] 引入类LSM索引结构，将近期更新放在快速内存层中处理，再逐步合并到磁盘图结构中，从而最大限度地减少块重写并保持磁盘布局局部性。DGAI [34] 采用解耦磁盘布局，因此更新邻接列表或向量数据时，无须重写整个页面。

2.2.3 基于树的方法。这些方法通过目录结构将向量组织到SSD块中，例如B^+树、学习型索引或基于哈希的分区树。高维向量首先被映射为一维键，通常采用局部敏感哈希、学习型模型或基于量化的编码，然后将这些键存储在有序磁盘块中。查询处理只需探测少量相关块，从而避免大规模倒排列表扫描。现有方法可从三个主要组成部分来理解：(i) 面向块的划分与布局，决定如何将向量转换为键并组织到SSD块中；(ii) 块过滤与查询执行，决定系统在搜索过程中如何选择并探测相关块；(iii) 更新与维护机制，处理索引如何在插入操作、删除操作和数据分布变化下保持高效。

面向块的划分与布局。基于将向量映射为一维键这一通用思路，不同方法采用各自的策略来构建键并组织

固态硬盘块，以实现高效访问。I-LSH [31] 根据 LSH桶标识符生成键，并将向量组织到由 B^+-树索引的桶中，每个桶都存储在一段连续的块范围内。PQBF [33] 使用 PQ编码计算距离的紧致下界，并组织固态硬盘块，使属于相似粗粒度分区的 PQ编码向量彼此相邻。查询时，系统按照 PQ 估算的向量与查询之间的距离对块进行排序，并按距离顺序探测，从而无需扫描所有块即可提前剪枝。学习函数方法 [26] 训练回归模型，将向量映射为更好地保留全局顺序的单调键，随后将固态硬盘块物化为局部性较高的有序运行段。

块过滤与查询执行。查询处理首先将查询向量映射到相同的键空间，并找出有希望探测的块范围。I-LSH [31] 在查询哈希得到的键值周围进行概率范围探测，而 EI-LSH [32] 则引入了具有提前终止机制的增量探测，逐步扩大搜索半径，直至找到足够的候选项。PQBF [33] 根据乘积量化近似距离对块排序，从而改进块过滤，使探测按距离顺序进行，而非完全依赖键范围。此外，近期方法还结合批量 I/O、预取和计算与 I/O 重叠，以降低延迟并摊销固态硬盘访问成本。

更新与维护机制。基于树的 VS 方法天然支持通过底层 B^+-树或学习型目录结构进行更新：插入操作会将新向量放入相应的键范围，删除操作则会移除或标记条目，而无需重构整个索引。不过，频繁更新可能导致块不平衡、键倾斜或局部性下降。EI-LSH [32]通过维护缓冲区并定期进行块合并来限制碎片化。数据分布发生漂移时，学习函数方法 [26] 依靠增量模型细化来维持映射准确率。使用基于乘积量化的分区时 [33], , 轻量级码本再训练或后台细化有助于保持索引质量。

2.3 弹性多层级向量搜索

随着向量数量的快速增长，企业级多层级向量搜索部署日益采用多层级存储架构，将高性能内存、大容量固态硬盘和高性价比对象存储相结合。这种分层设计使系统能够从数十亿扩展到数万亿个向量，同时平衡延迟、容量与成本效益。在本节中，我们介绍两种具有代表性的工业实践，即 Zilliz Cloud [17] 和 Turbopuffer [19], , 并讨论未来的研究方向。

Zilliz Cloud。 Zilliz Cloud（基于 Milvus 2.x 构建）采用按访问频率分层的内存—固态硬盘—对象存储架构。内存保存路由元数据、索引头和近期访问的向量，以支持超低延迟查找。固态硬盘持久化存储频繁查询的索引文件（例如 HNSW/倒排文件/乘积量化）和近期更新日志。对象存储作为弹性冷层，存储原始向量段和历史版本。查询采用感知分层的流水线：内存元数据定位目标索引分片，驻留在固态硬盘中的索引检索候选项，并按需获取存储在对象存储中的向量，再逐步将其提升到更快的层级。更新采用便于追加的模型，使用增量日志和墓碑标记，同时，后台压实会定期合并这些内容，以在无需完全重建的情况下保持索引质量。

TurboPuffer。 TurboPuffer 采用了类似的多层优化架构，面向超大规模向量存储库。热数据（最近插入的数据、活跃分区和路由结构）保留在内存中，而索引结构和温数据向量则存放在固态硬盘上，以支持高吞吐查询。冷分区和历史向量存储在云对象存储中（例如 S3/GCS），但仍可通过延迟分页和推测性预取进行访问。其更新流水线包含流式摄取、批量提交和惰性重组，以减轻碎片化并保持固态硬盘局部性。通过集成多层缓存、快照隔离和云原生持久性保证，TurboPuffer 实现了可扩展且具备成本效益的部署，同时拥有出色的写入性能。

未来展望。 尽管弹性多层向量搜索方法已在工业系统中展现出巨大潜力，但该领域仍处于发展的早期阶段。许多技术挑战仍有待解决：(1) 层感知索引协同设计，索引构建和查询策略必须共同考虑面向 SSD 和对象存储的数据布局以及层切换成本；(2)自适应与预测式缓存，利用访问预测主动迁移向量并降低延迟；(3) 对象存储高效查询，在访问冷数据时实现渐进式细化和最少的按需 I/O；(4) 弹性与自动扩展，支持动态工作负载下的在线重新分区和面向性能的层扩展；以及 (5) 成本优化，在云规模下最大限度提高成本效率，同时共同降低计算开销、SSD 写入和对象存储带宽消耗。

3 个人简介

Yitong Song 是香港浸会大学计算机科学系的博士后研究员。她的研究重点是向量数据库，并在 SIGMOD 和 VLDB 上发表了多篇论文。她将介绍异构存储 VS 技术。**Xuanhe Zhou** 是上海交通大学计算机科学系的助理教授。他的研究兴趣集中于数据系统。他曾获得 2025 年 SIGMOD Jim Gray 博士论文荣誉奖以及 VLDB 2023 年度最佳工业论文亚军。他曾在 SIGMOD、VLDB 和 ICDE 上开展教程。他将介绍内存驻留 VS 技术。

Christian S.Jensen 是丹麦奥尔堡大学的计算机科学教授。他的研究重点是时态、空间和多模态数据管理与分析。作为 ACM 会士和 IEEE 会士，他的 h 指数为 105，已发表 500 多篇论文，并获得了 2022 年 ACM SIGMOD 贡献奖和 2019 年 IEEE TCDE 影响力奖。他曾任 ACM TODS 和 The VLDB Journal 主编，并在 SIGMOD 和 VLDB 上发表过众多具有影响力的教程。他将介绍相关背景和研究机会。

Jianliang Xu 是香港浸会大学讲座教授兼计算机科学系主任。他的研究涵盖数据库、区块链和隐私感知数据管理。他是 IEEE 会士，h 指数为 65，已发表 300 多篇论文，并曾在 SIGMOD 和 ICDE 上作教程报告。他将介绍弹性多层向量搜索技术及开放问题。

参考文献

[1] FAISS. https://ai.facebook.com/tools/faiss。[2] Fabien André、Anne-Marie Kermarrec 和 Nicolas Le Scouarnec。2016。缓存局部性还不够：使用乘积量化快速扫描实现高性能最近邻搜索。VLDB 9, 4 (2016), 12。[3] Artem Babenko 和 Victor Lempitsky。2014年。用于极致向量压缩的加性量化。发表于 CVPR。931–938。[4] Artem Babenko 和 Victor Lempitsky。2014年。倒排多索引。IEEE TPAMI 37, 6 (2014), 1247–1260。[5] Erik Bernhardsson。2013。Annoy：近似最近邻，没错！ https://github.com/spotify/annoy。访问日期：2025-01-21。[6] Qi Chen、Bing Zhao、Haidong Wang、Mingqin Li、Chuanjie Liu、Zengzhong Li、Mao Yang 和 Jingdong Wang。2021年。SPANN：高效的十亿规模近似最近邻搜索。NeurIPS 34, 5199–5212。[7] Karima Echihabi、Kostas Zoumpatianos 和 Themis Palpanas。2021年。高维向量相似性搜索的新趋势：人工智能驱动、渐进式与分布式。PVLDB 14, 12 (2021), 3198–3201。[8] Simeon Emanuilov 和 Aleksandar Dimov。2024年。使用结合高级过滤的混合索引方法进行十亿规模相似性搜索。《控制论与信息技术》24, 4 (2024), 45–58。[9] Cong Fu、Chao Xiang、Changxu Wang 和 Deng Cai。2019年。使用导航式扩展图进行快速近似最近邻搜索。发表于 PVLDB，第 12 卷。VLDB Endowment，416–474。[10] Jianyang Gao 和 Cheng Long。2024年。RaBitQ：通过理论误差界对高维向量进行量化，以实现近似最近邻搜索。ACM SIGMOD 2, 3 (2024), 1–27。[11] Pengfei Gao、Zhao Tian、Xiangxin Meng、Xinchen Wang、Ruida Hu、Yuanan Xiao、Yizhou Liu、Zhao Zhang、Junjie Chen、Cuiyun Gao 等。2025年。Trae 智能体：一种采用测试时扩展的基于大语言模型的软件工程智能体。arXiv 预印本 arXiv:2507.23370 (2025)。[12] Tiezheng Ge、Kaiming He、Qifa Ke 和 Jian Sun。2013。用于近似最近邻搜索的优化乘积量化。发表于 IEEE TPAMI。2946–2953。[13] Siddharth Gollapudi、Neel Karia、Varun Sivashankar、Ravishankar Krishnaswamy、Nikit Begwani、Swapnil Raz、Yiyong Lin、Yin Zhang、Neelam Mahapatro、Premku- mar Srinivasan 等。2023年。Filtered-DiskANN：使用过滤器进行近似最近邻搜索的图算法。发表于 WWW。3406–3416。[14] Long Gong、Huayi Wang、Mitsunori Ogihara 和 Jun Xu。2020年。iDEC：用于近似最近邻搜索的可索引距离估计编码。PVLDB 13, 9 (2020)。[15] Hao Guo 和 Youyou Lu。2025年。通过将最佳优先搜索算法与固态硬盘对齐，实现低延迟基于图的向量搜索。第 19 届 USENIX 操作系统设计与实现研讨会（OSDI 25）（2025），171–186。[16] Ruiqi Guo、Philip Sun、Erik Lindgren、Quan Geng、David Simcha、Felix Chern 和 Sanjiv Kumar。2020年。利用各向异性向量量化加速大规模推理。ICML (2020), 3887–3896。[17] Cal Huang。2025年。从 S3 Vectors 迁移到 Zilliz Cloud：释放分层存储的强大功能。
h
t
t
p
s://
z
i
l
l
i
z.
c
o
m/
blog/
migrating-from-s3-vectors-to-zilliz-cloud-unlocking-the-power-of-tiered-storage 博客文章，Zilliz。[18] Qiang Huang、Jianlin Feng、Yikai Zhang、Qiong Fang 和 Wilfred Ng。2015年。用于近似最近邻搜索的查询感知局部敏感哈希。PVLDB 9, 1 (2015), 1–12。[19] Turbopuffer Inc. 2025年。Turbopuffer——基于对象存储、从基本原理构建的无服务器向量与全文搜索。https://turbopuffer.com/ 公司网站／产品页面。[20] Suhas Jayaram Subramanya、Fnu Devvrit、Harsha Vardhan Simhadri、Ravishankar Krishnawamy 和 Rohan Kadekodi。2019年。DiskANN：在单节点上进行快速、精准的十亿点级最近邻搜索。NeurIPS 32 (2019)。[21] Herve Jegou、Matthijs Douze 和 Cordelia Schmid。2010年。用于最近邻搜索的乘积量化。IEEE TPAMI 33, 1 (2010), 117–128。[22] Dingyi Kang、Dongming Jiang、Hanshen Yang、Hang Liu 和 Bingzhe Li。2025年。采用贡对齐图的可扩展基于磁盘的近似最近邻搜索。arXiv 预印本 arXiv:2509.25487 (2025)。[23] K Krishna 和 M Narasimha Murty。1999。遗传 K 均值算法。IEEE《系统、人类与控制论汇刊》，B 部分（控制论）29, 3 (1999), 433–439。[24] Huiling Li 和 Jianliang Xu。2025年。BAMG：用于基于磁盘的近似最近邻搜索的块感知单调图索引。arXiv 预印本 arXiv:2509.03226 (2025)。[25] Jinfeng Li、Xiao Yan、Jian Zhang、An Xu、James Cheng、Jie Liu、Kelvin KW Ng 和 Ti-chung Cheng。2018年。一种通用且高效的学习式哈希查询方法。发表于 ACM SIGMOD。1333–1347。[26] Mingjie Li、Ying Zhang、Yifang Sun、Wei Wang、Ivor W Tsang 和 Xuemin Lin。2020年。基于学习得到的函数实现高效输入/输出的近似最近邻搜索。IEEE ICDE (2020), 289–300。[27] Wen Li、Ying Zhang、Yifang Sun、Wei Wang、Mingjie Li、Wenjie Zhang 和 Xuemin Lin。2019年。高维空间中的近似最近邻搜索 数据——实 p验 , analyses, 与改 p进。*IEEETKDE* 32 ,(8.1475–1488.[28] Zhaoheng Li、Wei Ding、Silu Huang、Zikang Wang、Yuanjin Lin、Ke Wu、Yongjoo Park 和 Jianjun Chen。2025年。云原生向量搜索：全面的性能分析。arXiv 预印本 arXiv:2511.14748（2025年）。[29] Anqi Liang、Pengcheng Zhang、Bin Yao、Zhongpu Chen、Yitong Song 和 Guangxu Cheng。2025年。UNIFY：用于范围过滤近似最近邻搜索的统一索引。PVLDB（2025年）。[30] Hongfu Liu、Ziming Huang、Qi Chen、Mingqin Li、Y. Fu 和 Lintao Zhang。2018年。灵活平衡约束下的快速聚类。载于《2018年 IEEE 国际大数据会议（Big Data）》。[31] Wanqi Liu、Hanchen Wang、Ying Zhang、Wei Wang 和 Lu Qin。2019年。I-LSH：高维空间中 I/O 高效的 c-近似最近邻搜索。IEEE ICDE（2019年），1670–1673。[32] Wanqi Liu、Hanchen Wang、Ying Zhang、Wei Wang、Lu Qin 和 Xuemin Lin。2021年。EI-LSH：由提前终止驱动的 I/O 高效增量式 c-近似最近邻搜索。VLDBJ 30（2021年），215–235。[33] Yingfan Liu、Hong Cheng 和 Jiangtao Cui。2017年。PQBF：通过乘积量化实现 I/O 高效的近似最近邻搜索。CIKM （ 2017 年），667–676。[34] Jiahao Lou 、Quan Yu、Shufeng Gong、Song Yu、Yanfeng Zhang 和 Ge Yu。2025年。DGAI：用于高效更新与查询的解耦式磁盘图式 ANN 索引。arXiv 预印本 arXiv:2510.25401（2025年）。[35] Kejing Lu、Mineichi Kudo、Chuan Xiao 和 Yoshiharu Ishikawa。2021 年。HVS：基于沃罗诺伊图的分层图结构，用于解决近似最近邻搜索问题。PVLDB 15，2（2021年），246–258。[36] Vasilis Mageirakos、Bowen Wu 和 Gustavo Alonso。2025年。向量搜索索引裂解。arXiv 预印本 arXiv:2503.01823（2025年）。[37] Mikko I Malinen 和 Pasi Fränti。2014年。用于聚类的平衡 k 均值。载于《模式识别统计技术与结构句法模式识别联合 IAPR 研讨会（SPR 和 SSPR）》。Springer，32–41。[38] Yu A Malkov 和 Dmitry A Yashunin。2018年。利用分层可导航小世界图实现高效、稳健的近似最近邻搜索。IEEE TPAMI 42，4（2018年），824–836。[39] Jason Mohoney、Devesh Sarda、Mengze Tang、Shihabur Rahman Chowdhury、Anil Pacaci、Ihab F Ilyas、Theodoros Rekatsinas 和 Shivaram Venkataraman。2025年。Quake：向量搜索的自适应索引。arXiv 预印本 arXiv:2506.03437（2025年）。[40] Marius Muja 和 David G Lowe。2014年。面向高维数据的可扩展最近邻算法。IEEE TPAMI 36，11（2014年），2227–2240。[41] Jiongkang Ni、Xiaoliang Xu、Yuxiang Wang、Can Li、Jiajie Yao、Shihai Xiao 和 Xuecang Zhang。2023年。DiskANN++：利用查询敏感入口顶点，在同构映射图索引上进行高效的基于页的搜索。arXiv 预印本 arXiv:2310.00402（2023年）。[42] Zhaojie Niu、Xinhui Tian、Xindong Peng 和 Xing Chen。2025年。BlendHouse：ByteHouse 中的云原生向量数据库系统。IEEE ICDE（2025年），4332–4345。[43] James Jie Pan、Jianguo Wang 和 Guoliang Li。2024年。向量数据库管理技术与系统。载于《2024年国际数据管理会议伴随会议》。597–604。[44] Yun Peng、Byron Choi、Tsz Nam Chan、Jianye Yang 和 Jianliang Xu。2023年。多维数据库中的高效近似最近邻搜索。ACM SIGMOD 1，1（2023年），1–27。[45] Jeffrey Pound、Floris Chabert、Arjun Bhushan、Ankur Goswami、Anil Pacaci 和 Shihabur Rahman Chowdhury。2025年。MicroNN：设备端磁盘驻留可更新向量数据库。载于《2025年国际数据管理会议伴随会议》。608–621。[46] Aditi Singh、Suhas Jayaram Subramanya、Ravishankar Krishnaswamy 和 Har-sha Vardhan Simhadri。2021年。FreshDiskANN：用于流式相似性搜索的快速、准确的基于图的 ANN 索引。arXiv 预印本 arXiv:2105.09613（2021年）。[47] Yitong Song、Pengcheng Zhang、Chao Gao、Bin Yao、Kai Wang、Zongyuan Wu 和 Lin Qu。2025年。TRIM：通过增强的基于三角不等式的剪枝，加速高维向量相似性搜索。ACM SIGMOD（2025年），1–26。[48] Yifang Sun、Wei Wang、Jianbin Qin、Ying Zhang 和 Xuemin Lin。2014年。SRS：利用微型索引，在高维欧几里得空间中求解 c-近似最近邻查询。PVLDB（2014年）。[49] Yufei Tao、Ke Yi、Cheng Sheng 和 Panos Kalnis。2010年。高维空间中高效、准确的最近邻与最近点对搜索。ACM TODS 35，3（2010年），1–46。[50] Mengzhao Wang、Weizhi Xu、Xiaomeng Yi、Songlin Wu、Zhangyang Peng、Xi-angyu Ke、Yunjun Gao、Xiaoliang Xu、Rentong Guo 和 Charles Xie。2024年。Starling：面向数据段上高维向量相似性搜索的 I/O 高效磁盘驻留图索引框架。ACM SIGMOD 2，1（2024年），1–27。[51] Mengzhao Wang、Xiaoliang Xu、Qiang Yue 和 Yuxiang Wang。2021年。基于图的近似最近邻搜索：全面综述与实验比较。arXiv 预印本 arXiv:2101.12631（2021年）。[52] Yichuan Wang、Shu Liu、Zhifei Li、Yongji Wu、Ziming Mao、Yilong Zhao、Xiao Yan、Zhiying Xu、Yang Zhou、Ion Stoica 等。2025年。LEANN：低存储

向量索引。*arXiv* 预印本 *arXiv:2506.08276*（2025年）。[53] Chuangxian Wei、Bin Wu、Sheng Wang、Renjie Lou、Chaoqun Zhan、Feifei Li 和 Yuanzhe Cai。2020年。Analyticdb-v：面向结构化和非结构化数据查询融合的混合分析引擎。*PVLDB* 13，12（2020年），3152–3165。[54] Qian Xu、Juan Yang、Feng Zhang、Junda Pan、Kang Chen、Youren Shen、Amelie Chi Zhou 和 Xiaoyong Du。2025年。Tribase：一种利用三角不等式实现可靠、无损剪枝压缩的向量数据查询引擎。*ACM SIGMOD* 3，1 2025 1 28( )，–。[55] Yuming Xu、Hengyu Liang、Jin Li、Shuotao Xu、Qi Chen、Qianxi Zhang、Cheng Li、Ziyue Yang、Fan Yang、Yuqing Yang 等。2023年。SPFresh：面向十亿级向量搜索的增量原位更新。*SOSP*（2023年），545–561。[56] Peiqi Yin、Xiao Yan、Qihui Zhou、Hui Li、Xiaolu Li、Lin Zhang、Meiling Wang、Xin Yao 和 James Cheng。2025年。Gorgeous：重新审视磁盘驻留高维向量搜索的数据布局。*arXiv* 预印本 *arXiv:2508.15290*（2025年）。[57] Ziyang Yue、Bolong Zheng、Ling Xu、Kanru Xu、Shuhao Zhang、Yajuan Du、Yunjun Gao、Xiaofang Zhou 和 Christian S Jensen。2025年。明智地选择边：基于磁盘的近似最近邻搜索的单调路径感知图布局优化。*PVLDB* 18，11（2025年），4337–4349。[58] Channy Yun。2025年。推出 *Amazon S3 Vectors*：首个大规模支持原生向量的云存储服务（预览版）。https://aws.amazon.com/cn/blogs/aws/introducing-amazon-s3-vectors-first-cloud-storage-with-native-vector-support-at-scale/ 博客文章，AWS 新闻博客。[59] Cheng Zhang、Jianzhi Wang、Wan-Lei Zhao 和 Shihai Xiao。2025年。基于扩展邻域图的高效磁盘最近邻搜索。*ACM SIGIR*（2025年），2513–2523。[60] Minjia Zhang 和 Yuxiong He。2018年。Zoom：基于固态硬盘的向量搜索，以优化准确率、延迟和内存。*arXiv* 预印本 *arXiv:1809.04067*（2018年）。[61] Xi Zhao、Yao Tian、Kai Huang、Bolong Zheng 和 Xiaofang Zhou。2023年。面向高维空间的高效索引构建与近似最近邻搜索。*PVLDB* 16，8（2023年），1979–1991。[62] Shurui Zhong、Dingheng Mo 和 Siqiang Luo。2025年。LSM-VEC：用于动态向量搜索的大规模基于磁盘的系统。*arXiv* 预印本 *arXiv:2505.17152*（2025年）。