近似最近邻搜索技术综述

牟凌岳 [1)]

[1)](华中科技大学 计算机科学与技术学院, 武汉 430074)

摘 要 数字经济时代，大数据、云计算、移动互联网等新一代的技术催生出了海量的图像、视频、文本等非结构化数据。为了检索这些非结构化数据，通常使用深度学习技术将其转换为结构化向量，再进行向量检索。现在，基于近邻图的方法因其优秀的检索能力，成为了向量检索的主流算法。而精确搜索的代价极其昂贵，在对时延要求很高的场景下几乎不可能实现，所以催生了近似最近邻搜索技术。本文首先分析了当前向量检索的四种主流算法方向，之后又详细介绍了 DiskANN 和 SPANN 两种主流的向量检索算法，最后对向量检索技术做了一下简单的总结。

关键词 人工智能；向量检索

An Overview of Approximate Nearest Neighbor Search

Lingyue Mu[1)]

[1)](School of Computer Science & Technology, Huazhong University of Science and Technology, Wuhan 430074)

Abstract In the era of digital economy, new generation technologies such as big data, cloud computing and mobile Internet have spawned massive unstructured data such as images, videos and texts. In order to retrieve this unstructured data, deep learning techniques are often used to convert it into structured vectors, and then vector retrieval is performed. Nowadays, the method based on nearest neighbor graph has become the mainstream algorithm of vector retrieval because of its excellent retrieval ability. However, the cost of accurate search is extremely expensive, and it is almost impossible to achieve in the scenario with high delay requirements, so the approximate nearest neighbor search technology is born. This paper first analyzes four mainstream vector retrieval algorithms, then introduces DiskANN and SPANN two mainstream vector retrieval algorithms in detail, and finally makes a brief summary of vector retrieval technology.

Key words AI；Vector retrieval

1 引言

1.1 背景

随着移动智能设备的普及，信息在互联网上呈爆发式增长。近 10 年来，随着移动互联网的发展，个人可以随时使用移动设备接入互联网，进行文字、图片、视频的浏览，而伴随着的便是信息的几何式增长，预计在 2024 年，互联网上的数据规模将达到 149ZB[1]。因此，为了更快的对用户的信息搜索需求做出反应，如何在短时间内快速地在海量数据中进行匹配搜索，对研发人员提出了巨大的挑战。

为了搜索这些海量的信息，现代搜索引擎的策略是：在收集各类网络上的 数据后，使用机器学习技术将其编码为高维向量[2]。当用户在实际的搜索信息时，这些搜索原始数据的操作可以转化为在这些高维向量上进行。而转化为向量后，对这些向量使用各种策略进行相似结果搜索的问题，学界又将其称为向量检索问题[3]。

为了解决这一问题，最早采用的思路即暴力遍历[4]。它需要将查询点与数据库中所有数据点进行相似度计算，从而找到与查询点最为相似的数据点。然而，对于应用场景中的用户来说，查询的等待时间是有限的，因此在爆发性增长的数据量背景下，基于暴力遍历思路的精确查询所花费是不可接受的。因此，为了在搜索时间和搜索性能上达到平衡，目前主要的向量检索策略为近似最近邻搜索

最终修改稿日期：2023年12月27日；牟凌岳，男，2001年生，华中科技大学硕士在读，主要研究领域为计算机系统结构、向量检索. E-mail: m202373983@hust.edu.cn

(Approximate Nearest Neighbor Search，ANNS)[5]。

近似最近邻搜索的根本宗旨，即牺牲一定的查询准确度，来换取更快的查询速度。当前的常用的近似最近邻搜索方法有基于树、基于哈希、基于量化和基于近邻图四种类型。其中，基于近邻图的方法因为其优秀的检索能力，成为了目前向量检索方向的主流算法。

然而，目前基于近邻图的方法过于依赖内存。传统基于图的近似最近邻算法中，由于需要存储全部的原始向量数据以及相应的索引，基于图的算法需要消耗大量的内存。为了解决这一问题，微软提出了一种基于硬盘的图搜索算法 DiskANN[6-8]，它通过将大量占用内存的近邻图索引存储在固态硬盘(Solid State Drive，SSD)上，来降低检索时的内存占用。

1.2 研究现状

目前主流的向量检索算法往往都分为两个阶段，即建立索引阶段和搜索阶段。在算法的一开始，首先是使用额外的数据结构来作为辅助，通常称这种数据结构为索引，而在搜索阶段，则根据已经建立好的索引结构，使用查询数据在索引上进行距离计算来得到查询点的近似的最近邻居。

同时，这些向量检索算法又分为四大流派：1.基于树结构的方法。2.基于哈希的方法。3.基于量化的方法。4.基于近邻图结构的方法。其中，使用近邻图结构的方法，由于其较高的搜索速度和搜索准确度，是目前主流的向量检索算法。

然而，随着向量规模的快速增长，向量检索算法也面临着昂贵的内存成本。当前，超大规模向量检索相关研究可划分为两大类，一类是量化编码并辅助倒排索引策略的方案，另一类为近邻图加硬盘的优化方案。然而，由于量化编码策略的本质其实就是通过将大量数据进行了数据压缩，来减少索引体积和提高搜索速度，其中大量的原数据信息会出现巨大损失，与近邻图算法相比其搜索精度较低，在常规搜索场景下，用户要求一定搜索准确度时是不合适的。而如果只使用近邻图，直接对大规模向量进行索引构建，基于近邻图算法会消耗大量内存。例如，如果全部的构建过程全部在内存中进行，DEEP1B 数据集每个向量需要 384 字节，并且其有十亿个向量，只是读入原始到内存并构建基本的图结构都需要超过 350GB 的空间。

因此，使用近邻图算法，同时通过硬盘辅助的混合型向量搜索策略日益得到广泛的研究关注。其中，微软的 Suhas Jayaram 等人[6-8]首先提出的一种基于硬盘的图搜索算法 DiskANN，用于解决在上亿的大规模数据下图算法内存不足的问题，从而达到在上亿数据上满足高准确度的应用需求。该论文设计了一种索引常驻 SSD 的方案，基于论文中提出的图索引：Vamana，构建子图索引，并合并子图从而获得最终的合并索引。其整体的构建思路是：先构建基于量化的快速定位索引并保存在硬盘上，再将整个输入的数据进行分治构建图索引，最后合并子图得到最终的索引，将其保存在硬盘上。在搜索过程中，先通过驻留在内存中的量化编码索引距离在候选起点中寻找离查询点最近的候选节点，然后由候选起点开始进行搜索，通过在由 Vamana 图合并构建得到的索引上进行搜索，得到查询点的近似最近邻居。

后续，基于这种使用近邻图构建索引，并辅助硬盘等设备进行混合存储的优化策略。Qi Chen 等[9]提出了一种搜索算法 SPANN，其在建立索引时，将相似的向量小规模的聚类在一块存储于硬盘上，通过在搜索时只装载一定个数的聚类块来减少硬盘的访问次数。

类似的，也有一些研究选择使用比硬盘更加快的硬件设备来加速搜索速度，如 Liu Wei 等[10]提出的算法 HM-ANN，其构建索引的核心算法取材于经典的近邻图算法 HNSW，但是其使用了英特尔的傲腾设备作为其硬盘部分的替代品，由于傲腾作为一种非易失性内存，其读写速度高于传统固态硬盘，同时相对于内存其成本更加低廉，其在硬件速度上优于当前的固态硬盘。然而，其问题也很明显，首先在 2022 年，英特尔宣布关闭了傲腾的相关业务，即傲腾的后续硬件生产也将停止，同时，HM-ANN 需要对向量数据进行复杂的预处理，因此它的预处理复杂度较高。

同时，一些其他的研究侧重于利用其他硬件，如 GPU 和 FPGA 的高速并行性来提升算法的速度。如 Hongwu Peng 等人[11]提出的算法基于 FPGA，其出发点在于 FPGA 的计算速度比传统的 CPU 快得多，同时 FPGA 具有很高的可扩展性，因此可以根据需要调整硬件设计以满足不断增长的数据处理需求，然而一旦 FPGA 设计完成，更改和更新可能需要重新设计整个系统。而 facebook 提出的 Faiss-GPU 算法[12]则通过 GPU 中大量的小核心，通过超高的并行度来提升构建和搜索的速度。然而不论是使用 FPGA 和 GPU，其都需要极高的硬件成本和开

发成本，与结构相对简单且常用的固态硬盘相比有着许多的限制。

综上所述，近邻图硬盘优化方案已成为超大规模向量检索的主流方案，目前已经得到了广泛的应用。

2 向量检索

2.1 向量检索问题定义

向量检索问题也被称作为近似最近邻搜索问题(Apprxoimate Nearest Neighbor Search)，它是指在一个尺度空间中搜索与查询点最近点的优化问题。近似最近邻搜索问题的定义[12]可以描述如下：

给定一个含有 n 个数据点的集合 P，其属于 d 维的欧几里得空间 E^d。对给定的查询点 q，在 P 中找到距离查询点 q 最近的点 p，其中 p，q 间的距离小于等于 q 到 P 内某点最近距离的 1+ϵ 倍。其中，查询点 q 不在点集 P 中。

2.2 向量检索算法介绍

在过去的数十年中，为了解决高维空间中海量数据的近似最近邻搜索问题，人们提出了多种算法与数据结构。根据这些方法所使用的索引结构以及数据的组织方式，这些方法可以大致分成以下四个大类：1.基于树结构的方法；2.基于哈希的方法；3.基于量化的方法；4.基于图结构的方法。

本文将介绍两种基于图的用于大规模向量检索的向量检索算法，DiskANN 和 SPANN。

2.2 大规模向量检索相关研究

2.3.1 DiskANN

DiskANN[6-8]为 Suhas 等人提出了一种基于磁盘的 ANN 方案，该方案可以在单个 64 G 内存和足够 SSD 的机器上对十亿级别的数据进行索引、存储和查询，并且能够满足大规模数据 ANNS 的三个需求： 高召回、低查询时延和高密度（单节点能索引的点的数量）。该文提出的方法做到了在 16 核 64G 内存的机器上对十亿级别的数据集 SIFT1B 建基于磁盘的图索引，并且 recall@1 > 95% 的情况下 qps 达到了 5000，平均时延不到 3ms。

许多应用需要快速在十亿级别数据规模上做基于欧几里得距离的近似查询，目前有两种主流的方法：

基于倒排表+量化的方法。缺点是召回率不高，因为量化会产生误差。虽然可以提高 topk 以改善召回率，但是相应的会降低 qps。

基于分治。将数据集分成若干个不相交的子集，每个子集建基于内存的索引，最后对结果做归并。这种方法比较耗内存和机器。例如对 100M, 128 维的 float 数据集上建 NSG 索引，假设出度上限为 50， 大约需要 75G 内存。因此处理十亿级别的数据就需要多台机器。在阿里巴巴淘宝的实际应用中，将 20 亿 128 维浮点数据分到 32 个分片中分别建 NSG 索引，recall@100 为 98% 的时延大约在 5ms。当数据规模增加到千亿级别时需要数千台机器。

以上两种方法的局限性在于太依赖内存。所以这篇论文考虑设计一种索引常驻 SSD 的方案。索引常驻 SSD 的方案主要面临的挑战是如何减少随机访问 SSD 的次数和减少发起 SSD 访问请求的数量。将传统的基于内存的 ANNS 算法放到 SSD 上的话平均单条查询会产生数百个读磁盘操作，这会导致极高的时延。

DiskANN 提出了能够有效支持大规模数据的常驻 SSD 的 ANNS 方案。该方案基于文中提出的另一个基于图的索引： Vamana。DiskANN 的主要贡献包括：

1.DiskANN 可以在一台 64G 内存的机器上对十亿级别的维度大于 100 的数据集进行索引构建和提供查询服务，并在单条查询 recall@1 > 95% 的情况下平均时延不超过 5ms。

2.提出了基于图的新索引 Vamana，该索引相比目前最先进的 NSG 和 HNSW 具有更小的搜索半径，这个性质可以最小化 DiskANN 的磁盘访问次数。

3.Vamana 搜索性能不慢于目前最好的图索引 NSG 和 HNSW。

4.DiskANN 方案通过将大数据集分成若干个相交的分片，然后对每个分片建基于内存的图索引 Vamana，最后将所有分片的索引合并成一个大索引，解决了内存受限的情况下对大数据集建立索引的问题。

5.Vamana 可以和现有的量化方法如 PQ 结合，量化数据可以缓存在内存中，索引数据和向量数据可以放在 SSD 上。

Vamana 算法的思路与 NSG 类似，主要区别在于裁边策略。准确的说是给 NSG 的裁边策略上加了一个开关 alpha。NSG 的裁边策略主要思路是：

对于目标点邻居的选择尽可能多样化，如果新邻居相比目标点，更靠近目标点的某个邻居，可以不必将这个点加入邻居点集中。也就是说，对于目标点的每个邻居节点，周围方圆 dist(目标点，邻居点）范围内不能有其他邻居点。这个裁边策略有效控制了图的出度，并且比较激进，所以减少了索引的内存占用，提高了搜索速度，但同时也降低了搜索精度。Vamana 的裁边策略其实就是通过参数 alpha 自由控制裁边的尺度。具体作用原理是给裁边条件中的 dist(某个邻居点，候选点） 乘上一个不小于 1 的参数 alpha，当 dist(目标点，某个候选点）大于这个被放大了的参考距离后才选择裁边，增加了目标点的邻居点之间的互斥容忍度。

Vamana 的建索引过程：

1.初始化一张随机图；

2.计算起点，和 NSG 的导航点类似，先求全局质心，然后求全局离质心最近的点作为导航点。和 NSG 的区别在于：NSG 的输入已经是一张近邻图了，所以直接在初始近邻图上对质心点做一次近似最近邻搜索就可以了。但是 Vamana 初始化是一张随机近邻图，所以不能在随机图上直接做近似搜索，需要全局比对，得到一个导航点，这个点作为后续迭代的起始点，目的是尽量减少平均搜索半径；

3.基于初始化的随机近邻图和步骤 2 中确定的搜索起点对每个点做 ANN，将搜索路径上所有的点作为候选邻居集，执行 alpha = 1 的裁边策略。这里和 NSG 一样，选择从导航点出发的搜索路径上的点集作为候选邻居集会增加一些长边，有效减少搜索半径。

4.调整 alpha > 1 重复步骤 3。因为 3 是基于随机近邻图做的，第一次迭代后图的质量不高，所以需要再迭代一次来提升图的质量，召回率很大程度上受这一步的影响。

如图 2.1 所示，DiskANN 用 200 个二维点模拟了两轮迭代过程。第一行是用 alpha = 1 来裁边，可以发现改裁边策略比较激进，大量的边被裁剪。经过放大 alpha，裁边条件放松后，有很多条边被加了回来，并且第二行最右这张图，即最终的图中，有很多长边被加了回来。这样可以有效减少搜索半径。

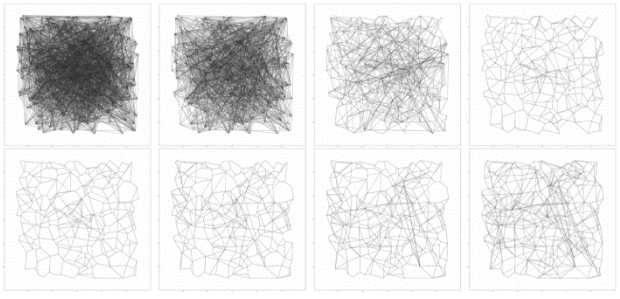

图 2.1 DiskANN 构建索引的过程

2.3.2 SPANN

2021 年，Qi Chen 等[9]人在 NeurIPS 提出了 SPANN，SPANN 为高效的内存-磁盘混合索引和搜索方案。

目前，大部分的 ANNS 方案主要基于向量数据能够完全放入内存中的假设，然而在实际业务中，有限的内存往往无法容纳日益增长的数据和索引文件。因此，向量检索引擎越来越青睐于使用外部存储设备与内存混合存储，从而满足容量、访问速度与召回率的平衡。

在设计上，SPANN 以较大粒度进行磁盘访问，充分利用了外部存储设备按块传输的访问特性，有效提高了外部存储设备的有效数据带宽。在十亿级向量数据检索场景中，SPANN 与现有的混合存储索引方案 DiskANN 相比，在内存开销相同的情况下，仅使用 1/3 到 1/2 的检索时间，即可达到相同的召回率。

SPANN 实现为倒排文件结构，倒排文件通过 K-Means 算法将向量检索引擎中的相互靠近的数据聚类到一起，并使用中心点来表示该聚类集合。查询向量在倒排文件的检索时，首先会找到与查询向量靠近的多个聚类集合，然后在多个聚类集合中进行进一步的搜索，因此能够避免对整个向量检索引擎中的数据进行搜索。每个聚类的中心点能够近似代表整个聚类集合中的向量，即如果查询向量离聚类的中心点很近，则可认为该查询向量离该聚类集合中所包含的所有向量都很近。因此，在倒排文件结构搜索过程中，查询点首先会和中心点进行一一比对，找到有限个相近的中心点向量，进一步对中心点向量所在的聚类进行搜索。相比于向量检索引擎的数据规模，中心点集合的规模要小得多。SPANN 把中心点向量集合常驻在内存中提供快速的聚类候选集合的检索，定位存储在磁盘中的大量的小规模聚类集合。进一步， SPANN 通过将存储在磁盘中的多个聚类集合加载到内存中进行搜索。为了提供高效的磁盘访问，SPANN 在倒排文件的建索引和检索过程进行了优化。对于倒排文件索引，增加聚类的个数，能够有效减少获得高召回所

需要搜索的向量数。SPANN 利用了这一现象，采用了多层级的负载均衡的聚类算法，生成大量的聚类集合来减少总的向量检索引擎的数据搜索数目。并且限制了每个聚类的规模，使每个聚类大小尽可能地均匀，这是为了最小化不同的查询向量的访问开销的方差。在先前的工作中，GRIP 通过实验分析验证了这一现象。GRIP 进行了在相同的数据集下采用不同的聚类集合数目构建索引的召回比较实验。图 2.2 为实验结果，其中 NC 为聚类集合的个数。从图中可以看出，在相同的搜素召回下，搜索聚类的数目并不随着索引生成的聚类数目线性相关。尤其在高召回的情况下，不同聚类数目的索引所搜索的聚类数目相近。同时，在相同搜索聚类个数的情况下，聚类数目大的索引的向量搜索总数比聚类数目小的索引要小得多。

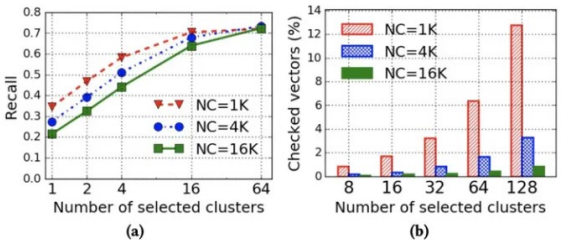

图 2.2 GRIP 实验结果

另一方面，SPANN 在对查询向量检索时实现动态剪枝，减少了磁盘访问的次数。因为查询向量具有“难易”之分，“难”的查询向量需要搜索大量的聚类才能获得高召回，“易”的查询向量需要搜索少量的聚类就能获得高召回。“易”的查询向量在搜索一定的聚类后，再增加搜索聚类的数目所获得的召回提高并不显著。SPANN 通过动态剪枝可以避免召回收益不高的聚类的查询。

由于较均匀地对向量检索引擎中的数据进行聚类会产生大量的小规模的聚类集合，使聚类间的边缘点增多，进而影响召回率。图 2.3 可以解释这个现象：图中存在两个聚类集合，在两个聚类集合之外的黄色点为查询向量，聚类内灰色点为中心点。当需要检索的聚类数有限，查询向量因为和绿色集合中的中心点更近，而选择绿色的聚类集合进行进一步的检索，实际上蓝色聚类集合中的红色边缘点里离询向量更近。针对磁盘访问优化策略所带来的召回率的损失，利用了磁盘的大容量，SPANN 对边缘点采用分配到多个聚类集合中的策略。

在实际应用中 SPANN 的磁盘访问优化方案和弥补召回下降策略面临如下的几个问题：（1）如何

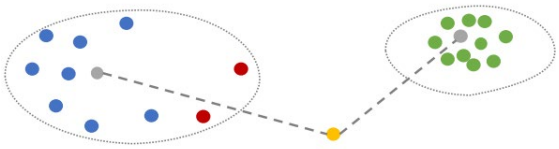

图 2.3 边界向量缺失

如图 2.1 所示，DiskANN 用 200 个二维点模划分？（2）向量冗余放置在多个聚类集合中所带来的额外的磁盘访问如何处理？（3）不同的查询向量需要查找的聚类集合个数不同，如何调整查询策略来应对“难”和“不难”的查询向量？SPANN 通过以下的三种关键技术来解决这些问题。

关键技术 1：均衡多层聚类算法

为了限制聚类集合的规模，SPANN 采用了多约束平衡聚类算法将数据集均匀地划分到大量的聚类集合中。损失函数表示为下述公式：

$$\min_{\mathbf{H},\mathbf{C}} ||\mathbf{X}-\mathbf{HC}||_{\mathrm{F}}^2+\lambda\sum_{i=1}^{N}(\sum_{l=1}^{|\mathbf{X}|}h_{li}-|\mathbf{X}|/N)^2,\ \text{s.t.}\sum_{i=1}^{N}h_{li}=1. \quad (1)$$

其中，X表示向量检索引擎中的向量矩阵，|X|表示向量检索引擎中的向量个数，C表示聚类中心点的矩阵，N表示聚类中心点的个数，H是表示向量划分的$|\mathrm{X}|*N$的位矩阵，λ 是控制聚类和条件约束比重的超参数。多约束聚类算法的训练过程和 K-means 的大致相同，所用的损失函数是在 K-means 的损失函数的基础上多增加了一个均衡划分的条件约束，采用的条件约束条件可以理解为最小化划分后的聚类大小的方差。

为了缓解训练大量聚类集合所带来的高昂的时间开销，采用了层次聚类的方法将时间复杂度从$O(|X|*dim*N)$降为$O(|X|*dim*k*log_k N)$，如图 2.4 所示。SPANN 采用自顶向下的层次聚类方式。层次聚类中的根节点会根据数据集中的所有向量采用较小的 K 值进行聚类，并把所有的向量划分到 K 个聚类中去。根节点和内部节点所生成的 K 个聚类作为它们的子节点。这 K 个子节点会根据其数据规模来区分是叶节点或是内部节点。如果节点中的数据集合大小小于特定的限制值，则作为落盘的叶子节点；否则作为下一层级的内部节点进行下一层聚类。反复迭代这个过程，直至所有的聚类集合都小于特定的限制值并落盘。同时，SPANN 为了快速地检索聚类的候选集，采用 SPTAG 索引结构维护存储在内存中的中心点。

为了缓解上述的边缘问题而导致的召回下降，SPANN 选择将聚类中的边缘点冗余放置到多个相

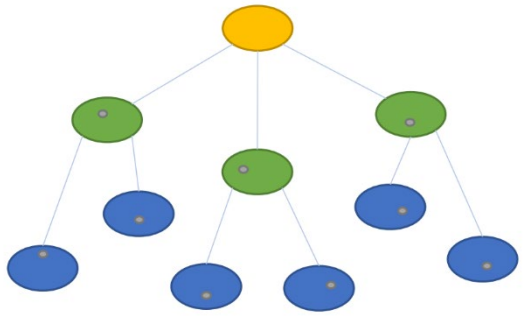

图 2.4 分层均衡聚类

近的聚类集合中，来增加这些向量的可见性。SPANN 将向量检索引擎中的向量分配到距离差不多的最近的几个聚类集合中，如图 2.5 所示，距离几乎相同这一条件能够筛选出边缘点而避免了向量检索引擎中所有向量的大量冗余所带来的高时延的磁盘访问开销。聚类中心点根据与向量检索引擎中的向量x的距离进行排序，向量x会被分配到中心点满足下述公式(2)的聚类集合中。公式(2)中使用了一个参数ϵ_1来限制边缘点重复放置的规模，SPANN 对边缘点根据与各个中心点的距离来筛选出在最小距离$Dist(x,c_{ij})$和$(1+\epsilon_1)$倍最小距离$(1+\epsilon_1)*Dist(x,c_{ij})$之间的中心点。

$$\mathbf{x}\in\mathbf{X}_{ij}\iff \mathrm{Dist}(\mathbf{x},\mathbf{c}_{ij})\le(1+\epsilon_1)\times\mathrm{Dist}(\mathbf{x},\mathbf{c}_{i1}),\quad \mathrm{Dist}(\mathbf{x},\mathbf{c}_{i1})\le\mathrm{Dist}(\mathbf{x},\mathbf{c}_{i2})\le\cdots\le\mathrm{Dist}(\mathbf{x},\mathbf{c}_{iK}) \tag{2}$$

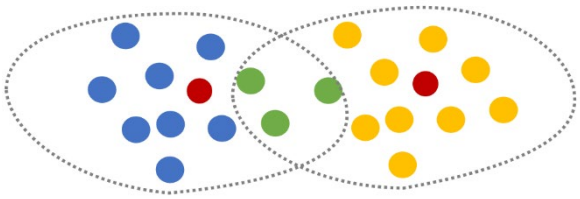

图 2.5 临近聚类分配

SPANN 采用 RNG 规则进一步减少边缘点的冗余放置。边缘点可以不重复放置在相近的聚类内，因为相近的聚类中心点相似度很高，被同时召回的概率也会提高，从而相同的数据会多次分配到磁盘中。对于聚类集合的中心点c_{ij}，如果x和c_{ij}的相对邻域内出现中心点c_{li}，中心点c_{li}和向量x的距离小于中心点c_{ij}和向量x的距离，则认为中心点c_{ij}和中心点c_{li}相近，向量x不会分配到c_{ij}的聚类中。

SPANN 采用查询感知的剪枝方法来调整不同查询向量需要加载的候选聚类的数目。通常，在倒排文件的检索过程中，首先会找到相同数量个最近的聚类中心点，然后对固定个聚类进行全量的搜索。然后，由于查询向量具有差异，有的“容易”查询向量只需要检索少数的聚类就能够获得高召回，而有的“难”查询向量需要检索更多的聚类。倒排文件使用相同的查找聚类个数会导致“容易”的查询向量需要检索很多召回收益不高的聚类。SPANN 通过查询向量与聚类中心点的距离远近程度来确定对该聚类的搜索是否是高召回收益的。SPANN 利用查询向量到与其最近的聚类中心点c_{i1}的距离$Dist(q,c_{i1})$作为尺度，使用搜索参数ϵ_2来控制动态剪枝的程度。当查询向量和某聚类中心点的距离大于$(1+\epsilon_2)*Dist(q,c_{i1})$，则认为是查询向量和中心点距离较远，对这一聚类进行进一步搜索的收益不高，可以进行剪枝，不对其进行搜索。

SPANN 基于倒排文件表索引进行了针对磁盘访问的数据布局和访问的优化方案设计。相比于 DiskANN，避免了向量量化所带来的数据精度损失和细粒度的磁盘随机访问，因此能够在较短的搜索时间内提供高召回。

3 总结与展望

本文首先介绍了向量检索的背景和问题的定义，接着对于各种算法的研究现状做了简要的叙述，指出目前主要有四个方向的向量检索算法：1.基于树结构的方法；2.基于哈希的方法；3.基于量化的方法；4.基于近邻图结构的方法。接着又详细介绍了两种用于大规模向量检索的典型算法，DiskANN 和 SPANN。这两种算法都是基于近邻图结构的算法，用廉价且大容量的磁盘代替内存来进行数据和索引的存储，解决了十亿规模下数据的存储问题。

随着人工智能大模型等领域的快速发展，向量形式的数据会越来越被人们所需要，如何在海量的数据中快速且准确地找到自己想要的数据是一个永恒的话题，有关于向量检索方向的研究会变得越来越多，相信在不远的将来，会出现更多形式的向量检索算法来满足人们的需要。

参 考 文 献

[1] Preeti et al. Chauhan. 2021. Big Data: Present and Future. Computer (2021).

[2] Fan Y, Xie X, Cai Y, et al. Pre-training methods in information retrieval[J]. Foundations and Trends® in Information Retrieval, 2022, 16(3): 178-317.

[3] Li W, Zhang Y, Sun Y, et al. Approximate nearest neighbor search on high dimensional data experiments, analyses, and improvement[J]. IEEE Transactions on Knowledge and Data Engineering, 2019, 32(8): 1475-1488.

[4] Dasgupta S, Sinha K. Randomized partition trees for exact nearest neighbor search[C]. Conference on learning theory. PMLR, 2013: 317-337.

[5] Hajebi K, Abbasi-Yadkori Y, Shahbazi H, et al. Fast approximate nearest-neighbor search with k-nearest neighbor graph[C]. Twenty-Second International Joint Conference on Artificial Intelligence. 2011.

[6] Jayaram Subramanya S, Devvrit F, Simhadri H V, et al. Diskann: Fast accurate billion-point nearest neighbor search on a single node[J]. Advances in Neural Information Processing Systems, 2019, 32.

[7] Singh A, Subramanya S J, Krishnaswamy R, et al. FreshDiskANN: A Fast and Accurate Graph-Based ANN Index for Streaming Similarity Search[J]. arXiv preprint arXiv:2105.09613, 2021.

[8] Jaiswal S, Krishnaswamy R, Garg A, et al. OOD-DiskANN: Efficient and Scalable Graph ANNS for Out-of-Distribution Queries[J]. arXiv preprint arXiv:2211.12850, 2022.

[9] Chen Q, Zhao B, Wang H, et al. SPANN: Highly-efficient Billion-scale Approximate Nearest Neighborhood Search[J]. Advances in Neural Information Processing Systems, 2021, 34: 5199-5212.

[10] Ren J, Zhang M, Li D. Hm-ann: Efficient billion-point nearest neighbor search on heterogeneous memory[J]. Advances in Neural Information Processing Systems, 2020, 33: 10672-10684.

[11] Peng H, Chen S, Wang Z, et al. Optimizing fpga-based accelerator design for large-scale molecular similarity search (special session paper)[C]. 2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD). IEEE, 2021: 1-7.

[12] Johnson J, Douze M, Jégou H. Billion-scale similarity search with gpus[J]. IEEE Transactions on Big Data, 2019, 7(3): 535-547.

附录.

课堂汇报记录：

问题：课堂汇报上所讲述的 VBASE 中定义的松弛单调性是基于什么原理？

松弛单调性是作者基于大量的实验观测得出来的结果，本身并没有严格的理论推导，是经过大量相同类型的实验由实验数据分析得出的结果，并把它应用在作者自己所构建的项目中，能够有效减少向量检索的迭代次数。

问题 2：论文中叙述的问题背景是否有主观性？

论文中指出，当前的向量检索无法实现对于标量的查询，即不能对向量中的每一个元素进行限制，需要先进行向量查询，再根据查询得到的结果进行筛选，之后才能得到想要的结果。整篇论文都是围绕着的这样一个背景来进行论述的。而实际上的情况是可以先对向量进行条件判断，之后再对满足条件的向量进行距离计算，把对标量的查询放在对向量距离计算之前，很显然文中并没有有关于这方面的叙述，这也是我认为本篇论文有所欠缺的地方。