# Towards Removing the Curse of Dimensionality (2012)

摘要：我们提出了两种解决高维空间中近似最近邻问题的算法。对于大小为 n 的数据集，存在于 R^d 中，这些算法需要的空间仅在 n 和 d 中是多项式级别的，同时实现了查询时间在 n 和 d 中是次线性的、在 d 中是多项式的。我们还展示了对其他高维几何问题的应用，如近似最小生成树。本文基于作者在STOC'98和FOCS'01论文中的材料，对这些论文的结果进行了统一、概括和简化。

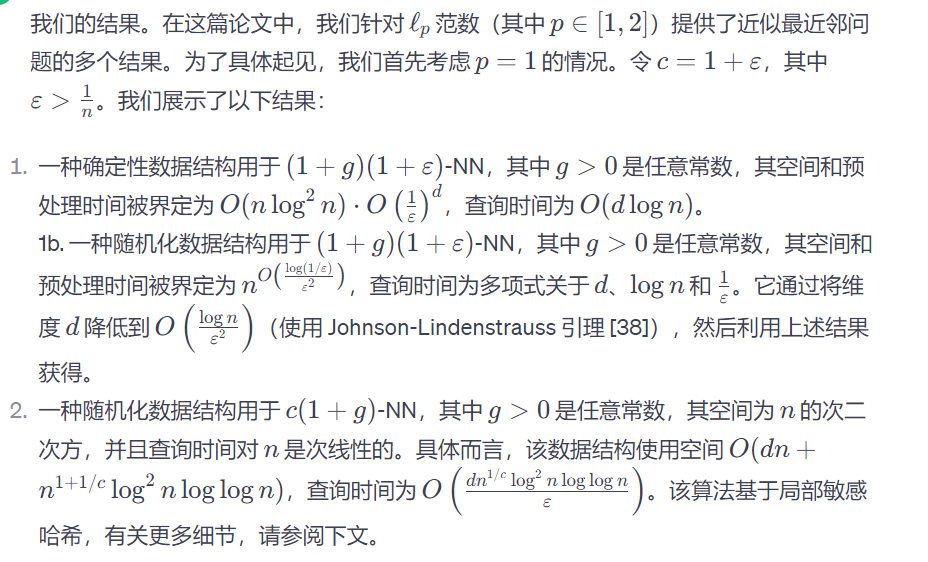
1. 介绍

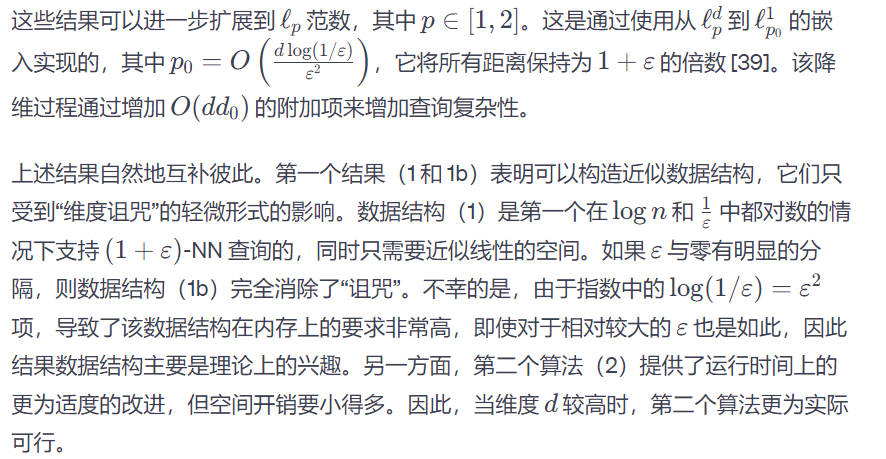
最近邻（NN）问题的定义如下：给定一个在已经定义了距离函数D的度量空间X上的n个点的集合P，对P进行预处理以高效地回答查找最接近查询点q（其中q属于X）的点的查询。一个特别有趣的情况是在d维欧几里德空间中，其中X = Rd，使用某个Lp范数。这个问题在许多领域中都非常重要，如数据压缩、数据库、数据挖掘、信息检索、图像和视频数据库、机器学习和信号处理。对这个问题的广泛兴趣源于它的广泛适用性。具体而言，许多大型数据集由可以表示为特征向量（即Rd中的点）的对象组成；在这种情况下，可以通过在特征空间中查找最近邻来找到与给定对象相似的对象。特征的数量（即维度）可以从几十到数百万不等。例如，可以将一个1000x1000的图像表示为1,000,000维空间中的一个向量，每个像素对应一个维度。

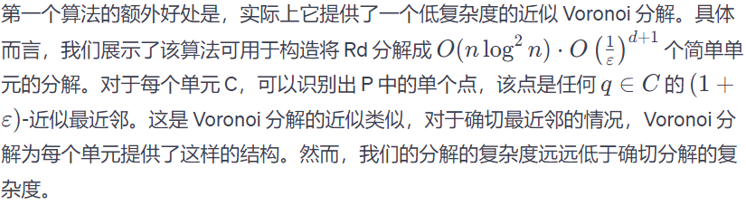
这个问题及其变种是计算几何学中的典型问题之一。最初在1960年代由明斯基和帕珀特提出 ([47]，第222页)，此后一直是大量研究工作的主题。对于点位于常数维度空间的情况，已经发现了许多高效的解决方案。例如，如果这些点位于平面上，最近邻问题可以在每次查询中以 O(logn) 的时间解决，只需要 O(n) 的存储空间 [59, 45]。

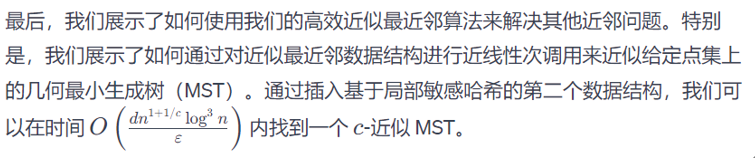
不幸的是，随着维度增加，这些算法变得越来越低效。更具体地说，它们的空间或时间要求在维度上呈指数级增长。特别地，最近邻问题有一个解决方案，具有 O(d^O(1) logn) 的查询时间，但使用了 n^O(d) 的空间 ([46]，基于 [22])。这部分原因是 P 的 Voronoi 分解，即 Rd 的分解成细胞，使得每个细胞内的所有点在 P 中都有相同的最近邻，其复杂性为 n^O(d)。另一方面，如果坚持使用线性（或近线性）的存储，即使对于随机点集，已知的最佳运行时间界也具有 min(2^O(d), dn) 的形式，这在适度的 d 下基本上是关于 n 的线性。更糟糕的是，维度对空间和/或时间的指数依赖关系（称为“维度诅咒”）在实践中也被观察到 [60]。

在无法消除对维度的指数依赖性的情况下，许多研究人员推测在维度足够大时这些问题没有有效的解决方案（例如，参见 [47]）。同时，这引发了一个问题，即是否可能在允许答案是近似的情况下消除对 d 的指数依赖性。具体而言，在 c-近似最近邻问题中，算法允许报告距离 q 最近的点 p，该算法允许报告任何在距离 q 到 p 的距离的 c 倍以内的点，其中近似因子 c > 1。这种方法的吸引力在于，在许多情况下，近似最近邻几乎和确切的最近邻一样好。特别是，如果距离度量准确捕捉到某种质量概念，那么距离的小差异不应该太重要。此外，可以使用高效的近似算法来解决确切的最近邻问题，只需枚举所有近似最近邻并返回遇到的最近点。









我们的技术：我们的近似最近邻数据结构是通过两个步骤获得的。首先，我们展示了如何解决近似最近邻问题的“决策版本”，我们称之为近似近邻问题。在这里，给定一个固定的半径r，目标是为给定点集合P 构建一个数据结构，对于任何查询点 q，它会执行以下操作：如果存在一个与 q 距离不超过 r 的点在 P中，那么它会报告一个在距离 cr 以内的点 p0∈P。我们展示了如何构建两种这样的数据结构。一种利用上述的近似 Voronoi 分解，具有快速的查找时间和在 d 上呈轻微指数增长的空间。第二种基于局部敏感哈希（LSH）的概念。其关键思想是以一种方式对点进行哈希，使得靠近彼此的对象发生碰撞的概率远高于相距较远的对象。然后，可以通过对查询点进行哈希并检索存储在包含该点的桶中的元素来确定近邻。我们展示了这样的哈希函数族存在于汉明距离及其变体中，并将其扩展到 ls 范数。

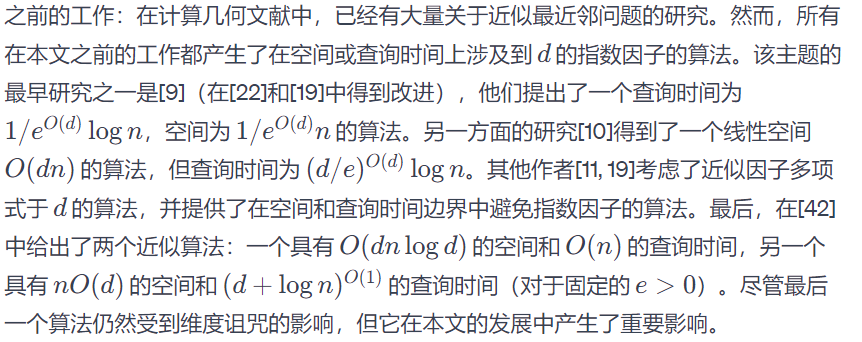
在第二步中，我们展示了如何将近似最近邻问题规约到近邻问题。执行此任务的一种简单方法是为后者的问题构建多个数据结构，其中半径为r = D, D/c, D/c^2,…，其中 D是查询点与点集之间的最大可能距离。然而，通常情况下，这种方法会导致空间复杂度不受n 的任何函数限制。我们通过提供一种更有效的规约来克服这个问题，该规约只使用一些半径r。我们的规约将近邻数据结构的空间复杂度乘以 O(log^2 n) 的因子，并将查询时间乘以 O(log n)。将规约与上述解决近似近邻问题的算法组合，得到了解决近似最近邻问题的算法。

与会议论文的关系：由于与该论文基础上的初版会议论文[35, 30]相比已经过去了一段时间，我们能够大大简化一些论证。因此，本文中的一些算法更简单，有时比原始论文中的算法更一般化。特别是，本文中提出的从近邻到最近邻的规约是对[30]中规约的简化（该规约本身比[35]中的规约简单且更有效）。它适用于一般的度量空间，并且可以使用近线性数量的近似近邻查询来执行。相比之下，[30]中的预处理算法是为 ls 空间定制的，而[35]中的算法在二次时间内运行。

然而，我们注意到我们的规约（以及[30]的规约）的一个副作用是，在近似近邻的实例中，点集的直径与搜索半径 r 的比率不再很小（对数多项式）。这个性质在设计l1d空间的高效 LSH 函数时是有用的。我们可以通过直接利用 Bern 的“随机分桶”方法[11]（引理 3.1）来确保一个更弱的性质。幸运的是，该性质对我们的目的已经足够。

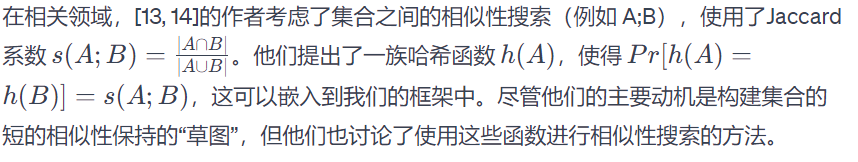
另一个经过实质性概括的结果是计算近似 MST 的算法。手稿[36]中概述的算法依赖于一种用于在更新过程中维护点对的近似接近的数据结构，该数据结构用于模拟Kruskal的MST算法。相反，在本文中，我们提出了从近似 MST 问题到动态近似近邻数据结构的一般规约，该规约适用于任意度量。规约仍然基于 Kruskal 算法。但是，它具有特别简单的形式，受到[12]算法的启发（请参阅下一小节的评论）。

**1.1相关工作**



本文中引入的局部敏感哈希方法在文献中有几个前辈，这些前辈研究了基于多索引哈希的算法，用于检索与汉明距离相关的向量对。尽管这些论文采用的分析框架使得结果通常无法与我们的结果相比，但一些见解是共享的。其中一些论文考虑了在数据集中找到所有“接近”点对的相关问题。为简单起见，我们将它们转化为近邻框架，因为它们可以通过执行基本上是 n 个单独的近邻查询来解决。

通常，哈希函数会将向量投影到一些坐标子集{1,…,d}上。在一些论文中[54, 29]，作者考虑了数据点在概率模型下均匀随机选择的情况，查询点是与数据集中的某个点“接近”的“随机”点。另一种方法[40]是假设数据集是任意的，但几乎所有点都远离查询点。最后，论文[17]提出了一种算法，不对输入做任何假设，并提供了一种确定参数（此处表示为 k 和 L）以实现所需的灵敏度和准确性水平的方法。



与我们的会议论文[35]齐头并进的是论文[43]，该论文提出了一个与我们结果（1b）相似的算法。具体而言，它提供了一种数据结构，对于d1范数的情况，可以在使用空间（dn）O(1/e2)的情况下实现O(d(logn+1/e)O(1))的查询时间。对于2范数，查询时间变为O(d2(logn+1/e)O(1))。他们的数据结构提供的概率保证略为更强：如果构造过程正确（以一定概率发生），则数据结构对所有查询q返回一个正确的近似最近邻居。虽然技术发展略有不同，但总体方法相似，即使用一种随机降维的形式将维数减小到O(logn/e2)。

在另一个并行发展中，论文[12]提出了一个MST（最小生成树）的（1+e）-近似算法（对于e < 1），其运行时间为O(dn1-ae2)（其中a > 0是某个绝对常数）。

进一步的发展 自从本论文的会议版本在[35, 36, 30]中出现以来，关于近似最近邻搜索的许多进一步发展已经出现。在本节中，我们简要讨论其中一些结果。关于该材料的更深入处理请参阅[34]。

从近似最近邻到近邻问题的归约以及由此产生的近似Voronoi分解在[7]中通过使用一种良好分离对分解[18]进一步改进，以生成构建近似Voronoi图所需的单元格（注意，与我们的构造不同，单元格的数量是与n线性相关，但与维度呈指数相关）。这个构造使用了与[30]中建议的相同框架。这一结果被扩展以改善在查询时间和使用的空间之间的与e的依赖关系的权衡。有关更多详细信息和相关工作，请参阅[8]。这个构造导致了一个复杂度为n/eO(d)的近似Voronoi图。

另一种构造方法在[57]中提出。借鉴了从最近邻搜索到近邻问题的归约建议（见[30]），他们通过对近邻问题进行轻微改变，将空间需求减少了对数因子。通过稍微改变近邻问题，他们将空间限制减小到线性，从而产生一个复杂度为n/eO(d)的近似Voronoi图。

算法（1b）在空间限制方面达到的指数较可能接近最优。具体而言，[6]表明在汉明空间中，任何进行恒定数量内存访问的（1+e）-近似数据结构都需要nW(1/e2)的存储空间（我们的数据结构只进行一次内存访问）。关于精确和近似最近邻问题的下界已经有了大量的研究，详见[34]。

在设计各种度量的LSH函数方面已经取得了实质性的进展，并且对它们的限制有了更深刻的理解，详见[4]。特别是对于l1范数，本文中给出的运行时间指数的1/c限界是紧致的（[50]，基于[48]）。有关更一般计算模型的下界，见[53]。相反，对于2范数，可以将指数进一步降低到1/c2（[3]，基于[26]）。此外，在[52]中提出了一种利用LSH函数的不同算法；这种数据结构实现了近似线性存储，但代价是将指数增加了一个常数因子。

从更一般的角度来看，对于高维l1空间存在快速近似最近邻算法使得可以通过将其他度量嵌入低失真的l1空间来解决这些问题。这种方法对于编辑距离的变体[49, 25, 24, 20, 51]、地球移动距离[21, 37, 28, 5]和其他度量已经被证明是有效的。对于这种嵌入的下界也进行了研究[41]。

最后，LSH算法及其变种已经成为在高维空间中进行相似性搜索的流行实用算法。它们已成功应用于各种领域的计算问题，包括网络聚类[33]、计算生物学[15, 16]、计算机视觉（参见[58]中的选定文章）、计算药物设计[27]和计算语言学[56]。

…

…

…

3.2 局部敏感哈希