基于容忍因子的近似最近邻混合查询算法

贺广福[1,2]，薛源海[1]，陈翠婷[1,2]，俞晓明[1,2]，刘欣然[1,3]，程学旗[1,2]
1. 中国科学院计算技术研究所，北京 100190；
2. 中国科学院大学，北京 101408；
3. 北京邮电大学，北京 100876

摘要
近似最近邻搜索（ANNS）是计算机领域中一种重要的高效相似度搜索技术，可用于在大规模数据集中进行快速信息检索。随着人们对高精度信息检索的需求不断增长，同时使用结构化信息和非结构化信息进行混合查询的方式也得到了广泛应用。然而，基于近邻图的过滤贪心算法在混合查询时可能会因结构化约束条件的影响导致连通性降低，进而损害搜索精度。为此，提出了一种基于容忍因子的过滤贪心算法，通过容忍因子控制不满足结构化约束条件的顶点参与路由，在不改变索引结构的前提下维持原有近邻图的连通性，克服了结构化约束条件对检索精度的负面影响。实验结果证明，新算法可以在不同结构化约束强度下实现ANNS的高精度搜索，同时保持检索效率。该研究解决了基于近邻图的ANNS在混合查询场景中的问题，为大规模数据集的快速混合查询信息检索提供了一种有效的解决方案。

关键词
混合查询；向量检索；最近邻搜索；过滤搜索

中图分类号：TP391.3，TP18　　文献标志码：A　　doi: 10.11959/j.issn.2096-0271.2024010

Approximate nearest neighbor hybrid query algorithm based on tolerance factor

HE Guangfu[1,2], XUE Yuanhai[1], CHEN Cuiting[1,2], YU Xiaoming[1,2], LIU Xinran[2,3], CHENG Xueqi[1,2]
1. Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China
2. University of Chinese Academy of Sciences, Beijing 101408, China
3. Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract
Approximate nearest neighbor search (ANNS) is an important technique in the field of computer science for efficient similarity search, enabling fast information retrieval in large-scale datasets. With the increasing demand for high-precision information retrieval, there is a growing use of the hybrid query method that utilizes both structured and unstructured information for querying, which has broad application prospects. However, filtered greedy algorithms based on nearest

neighbor graphs may reduce the connectivity of the graph due to the impact of structural constraints in hybrid queries, ultimately damaging search accuracy. This article proposes a tolerance factor based filtered greedy algorithm, which controls the participation of vertices that do not meet structural constraints in routing through a tolerance factor, maintaining the connectivity of the original nearest neighbor graph without changing the index structure, and overcoming the negative impact of structural constraints on retrieval accuracy. The experimental results demonstrate that the new method can achieve high-precision search for ANNS under different levels of structural constraints, while maintaining retrieval efficiency. This study solves the problem of ANNS based on nearest neighbor graphs in hybrid query scenarios, providing an effective solution for fast hybrid query information retrieval in large-scale datasets.

Key words
hybrid query, vector search, nearest neighbor search, filtered search

0 引言

近年，近似最近邻搜索（approximate nearest neighbor search，ANNS）已经在图像搜索[1-3]、信息推荐[3-4]、序列匹配[5]、大语言模型应用[6]等领域成为一个重点研究课题。基于近似近邻图的ANNS算法在处理大规模高维数据搜索时显示出较高的搜索效率和准确性[7]。随着人们对高精度信息检索的需求不断增长，同时使用结构化信息和非结构化信息进行混合查询（hybrid query）的方式也得到了广泛应用。例如在电商的商品搜索中，可能需要依据图片这类非结构信息和生产日期、价格、销量等结构化信息进行混合查询，精确地搜索相关产品。然而，ANNS主要关注非结构化信息，对结构化信息的影响关注较少，这导致ANNS在满足结构化约束信息检索需求方面的表现并不理想。

为了解决这类问题，研究者们在相关领域提出的混合查询方法一般可以分为3类[8]，即前处理（pre-process）、后处理（post-process）和内联处理（inline-processing）。其中前处理需要大量的时间和空间来构建数据结构[9-10]；后处理需要对检索的结果进行修正，存在一定的误差和不确定性[11]；内联处理方法可以直接在查询过程中进行最近邻的计算，具有高效、准确的特点，尤其是对实时性要求较高的应用场景[12-14]。

在非结构化信息检索中，基于近邻图类的ANNS算法在高维度、大规模数据集上表现优异，因此扩展此类方法的内联处理混合查询具有较好的应用前景，是ANNS领域的重要研究分支。基于近邻图的内联混合查询中，一般使用贪心过滤搜索（filtered greedy search，FGS）算法在近似近邻图上检索路由[8,12,15]。但是，该方法在查询时仅依据满足结构化约束的点进行路由，降低了ANNS在构建索引时近邻图的连通性预期，导致检索结果的精确度下降。

针对上述挑战，本文提出了一种基于容忍因子的过滤贪心搜索（tolerance factor based filtered greedy search，TF-FGS）算法。TF-FGS算法通过引入容忍因子，在不改变索引结构的前提下保留近邻图的连通性，从而解决由于结构化约束对搜索算法造成的影响，提升了结果准确率。同时，TF-FGS算法保留了与没有结构化约束的ANNS近似的搜索效率。实验表明，在混合查询场景中，本文提出的方法在不牺牲检索效率的前提下，有更好的搜索准确率。

本文的主要贡献如下。

- 分析了FGS算法存在的问题。在搜索过程中，路由候选集合所有向量必须满足结构化约束，过度约束了可路由的路径，影响了路由过程中近邻图的连通性，导致搜索准确率下降。
- 提出了一种TF-FGS算法，在不改变索引结构的条件下，允许不满足结构化约束的向量参与路由，在维持原有检索效率的同时，提升了检索结果的准确率。
- 在不同类型的基准数据集上进行了实验，验证本文提出的TF-FGS算法相比FGS算法，在保持查询效率不变的情况下，提高了混合查询的准确率。

1 问题定义

本节定义了最近邻搜索、近似最近邻搜索、混合场景下的近似最近邻问题。

1.1 最近邻搜索和近似最近邻搜索

最近邻搜索问题是依据查询条件 $\boldsymbol{x}_q \in \mathbb{R}^{\mathrm{d}}$ 和 $k \in \mathbb{N}$，从包含 N 个 $\boldsymbol{x} \in \mathbb{R}^{\mathrm{d}}$ 向量的数据集 P 中，搜索出离查询目标向量 $\boldsymbol{x}_q$ 最近的 k 个向量。距离的计算方式有多种[16-17]，不限于欧氏距离、余弦距离等。本文以欧氏距离为例进行说明，计算方法如下：

$$\mathrm{dist}(\boldsymbol{x}_p, \boldsymbol{x}_q) = \sqrt{\sum_{i=1}^{n} (\boldsymbol{x}_{pi} - \boldsymbol{x}_{qi})^2} \tag{1}$$

在实际应用中，需要在高维数据集中进行最近邻搜索。但是，高维数据距离计算开销很大，精确搜索的查询效率很低，无法满足实际需求。因此，一般设计近似最近邻搜索算法来解决最近邻搜索问题。ANNS算法的目标是最大化为：

$$\mathrm{precision}@k = \frac{|\mathrm{G} \cap \mathcal{C}|}{k} \tag{2}$$

高效地召回拥有 k 个向量的 $\mathcal{C}$ 集合，其中，G 表示在 P 中距离 $\boldsymbol{x}_q$ 最近的 k 个向量。

1.2 结构化约束

在数据集 P 中，向量 $\boldsymbol{x} \in P$ 包含了若干属性或者标签 $F_x \subseteq \mathcal{F}$，其中 $\mathcal{F}$ 表示向量上有限个标签或者属性。结构化约束条件为 $C = \phi(F_x)$，定义为由向量的属性或标签 $\mathcal{F}$ 构成的逻辑关系表，值为True（满足约束）或False（不满足约束）。结构化约束按照约束条件的不同，一般分为等值约束、多值约束和区间约束。

经过结构化约束后的向量集合为P_C，其满足：

$$P_C = \{\boldsymbol{x} \in P : C\} \tag{3}$$

不满足结构化约束的向量集合为：

$$\overline{P_C} = \{\boldsymbol{x} \notin P_C\} \tag{4}$$

可以使用不满足结构化约束的向量比例来表示结构化约束强度：

$$Q = \frac{|\overline{P_C}|}{N} \tag{5}$$

1.3 带结构化约束的近似最近邻搜索

给定查询条件为目标向量 $\boldsymbol{x}_q$，最近邻结果个数为 k，结构化约束为 C，带结构化约束的近似最近邻搜索需要从 P_C 中查询距离 $\boldsymbol{x}_q$ 近似最近的前 k 个向量。此时算法目标 $\mathrm{precision}@k$ 中的 G 为在 P_C 中距离 $\boldsymbol{x}_q$ 最近的 k 个向量。

2 相关工作

在ANNS算法上扩展结构化数据过滤是解决混合查询的重要方法。按照结构化约束相对向量检索召回的先后顺序，可将这些算法分为后处理、前处理和内联处理。其中基于近邻图索引结合过滤贪心搜索算法的内联混合查询（inline hybrid query）表现出较好的检索效率和较低的额外开销。

2.1 后处理类混合查询方法

最直接的方法是后处理类的混合查询方法：在查询时，先执行非结构化查询，然后应用非结构化约束信息来过滤非结构化查询召回的数据。例如，京东的Vearch[11]首先通过ANNS获取满足非结构化约束的候选集，然后在候选集上执行标签过滤，得到最终结果。这种方法很容易在原来的方法上扩展实现。但是，由于结构化约束是用户输入的查询条件，非结构化查询阶段召回的数据中是否有满足过滤条件的向量具有较大的不确定性，特别是对于严苛的过滤条件，会导致满足约束的向量很少，可能必须召回大量候选向量才能在后处理时召回期望的查询结果。

2.2 前处理类混合查询方法

与后处理的混合查询方法相反，有些研究者提出了前处理方法：用结构化约束属性构建额外的索引，召回满足过滤条件的向量，然后再使用原来的ANNS方法查询非结构数据。构建额外的索引结构往往需要高昂的计算或者存储代价。

Xu等人[18]提出的MA-NSW算法为每个属性构建一个近邻子图，形成一个多层次的索引结构，在执行查询时，先通过导航树映射到满足结构化约束的近邻子图，再在子图上执行ANNS获取满足结构化约束的混合查询结构。MA-NSW算法实现了高效的混合查询，但是在结构化属性比较多的情况下，其朴素地为每一条结构化数据组合构建子图的方式将导致巨大的时空构建复杂度，从而难以普及应用。Weaviate[10]和Milvus[9]均在原来的ANNS查询前，通过结构化约束过滤出符合条件的向量列表（approval list），而在非结构化查询时，仅考虑此列表中的向量。此向量列表在大规模检索中需要较大的额外内存开销。

2.3 内联处理类混合查询方法

有些算法可以在原来非结构查询的过程中动态地应用后处理方法，被称为内联处理方法。此类方法中的典型代表是FAISS-IVF[12]，它将向量索引和结构化约束属性同时进行倒排索引，在查询时同时筛选结构化和非结构化信息。Pinecone的混合搜索也利用该方法[13]。后续也有一些研究者提出了基于树和基于量化方法的混合查询方法，例如王梦召等[14]提出的NHQ方法。

2.4 基于近邻图的混合查询方法

在非结构化查询场景下，基于哈希的方法使用哈希函数将数据点映射到哈希桶中，从而实现近似最近邻搜索。哈希函数具有高效的查询速度，尤其适用于高维数据，但是哈希方法存在哈希冲突问题，即不同的数据点可能映射到相同的哈希桶中，导致查询结果不准确。基于树结构的方法，例如R-Tree[19]、KD-Tree[20-21]和Ball-

Tree[20,22]，通过递归地划分数据空间，可以快速定位最近邻，同时还可以有效地剪枝，减少搜索空间。但是在高维数据场景中，树结构很容易变得不平衡，导致搜索效率下降。

但是，基于近邻图的索引，如HNSW[23]、NSG[24]、SSG[25]和Vamana[15]，在针对特定召回率时，与KD-Tree、IVF、LSH等索引相比，效率提高了一个数量级[7]，而且随着数据规模的增加，这种差距只会更大。因此，扩展基于近邻图的ANNS索引支持混合查询，可以实现更高吞吐量和更优召回率。

前人的研究大多使用过滤贪心搜索算法来扩展基于近邻图的混合查询。如Jayaram等人[15]提出的DiskANN和Gollapudi等人[8]提出的FilteredDiskANN，均使用过滤贪心搜索算法实现混合查询。该方法会在搜索时依据结构化约束条件进行剪枝，可以快速收敛检索结果，具有优秀的检索性能。但是，这样做会破坏最近邻图的连通性假设，有一定的副作用。例如在过于严苛的结构化条件约束下，符合条件约束的向量很少，在路由时，要么无法到达最近邻，要么到达最近邻需要进行更多跳的搜索。

本文提出的算法在不改变索引结构的条件下，通过控制容忍因子来调节点 $\boldsymbol{x} \in \overline{P_c}$ 参与路由的比例，缓解结构化约束条件对近邻图连通性的影响，从而解决过滤贪心搜索算法的结构化约束问题。

3 过滤贪心搜索算法问题分析

本节首先进行了基于近邻图索引和过滤贪心搜索算法的混合查询实验。实验表明，大多数场景下，过滤贪心搜索算法有较高的检索准确率；但是在结构化约束条件过强时，可能会出现检索准确率下降的问题。然而，强结构化约束条件下的混合查询是重要的应用场景之一，为了进一步提升用户搜索体验，亟须解决该问题。本文从过滤贪心搜索算法的原理出发，分析了该问题的原因。

3.1 FGS混合查询实验

本文使用了基准数据集GloVe[26]中的GloVe.6B.50d进行实验，该数据集包含了来自维基百科2014和Gigaword5的400 000个词汇及其对应的向量。但是该数据集缺少数据属性或者标签 $\mathcal{F}$，本文随机生成了用于结构化约束的属性，并从数据集中随机选择了10 000条数据进行实验。

Erd□s等人[27]和Watts等人[28]的研究表明，近邻图的连通性可以使用顶点的平均度表示。顶点的度分布与约束条件强度的关系如**图1**所示，检索精确度与约束条件强度 Q 的关系如**图2**所示。当结构化约束强度 $Q<73\%$ 时，超过一半顶点的度在20%区间及以上，此时从**图2**可知，检索的 $\text{precision}@k$ 较高，检索结果绝大部分为相关数据，用户可以很快找到相关信息；当结构化约束强度 $Q \geqslant 83\%$ 时，绝大多数（超过90%）顶点的度在10%区间，顶点的度大量减少，近邻图的连通性下降，此时观察**图2**，检索的 $\text{precision}@k$ 极速下降，检索结果中包含的相关数据比例急剧下降，最终导致用户无法有效地找到相关信息。

从上述实验中可以得到以下结论。

- FGS算法在多数场景下能保证检索精确度。但是，在结构化约束强度 Q 过大时，检索准确率较低。特别是在极端情况下，检索准确率趋于0。
- 结构化约束强度 Q 与近邻图联通性成反相关。结构化约束强度 Q 越大，则搜索路由时索引的连通性越弱；反之，索引的

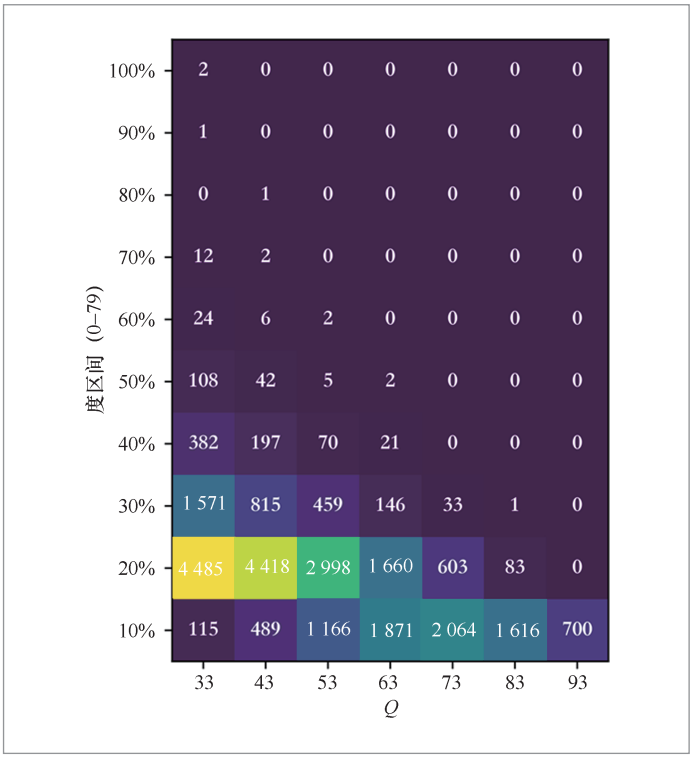

图 1 顶点的度分布与约束条件强度的关系
（纵坐标将顶点的度分为 10 个区间，统计每个区间的顶点个数）

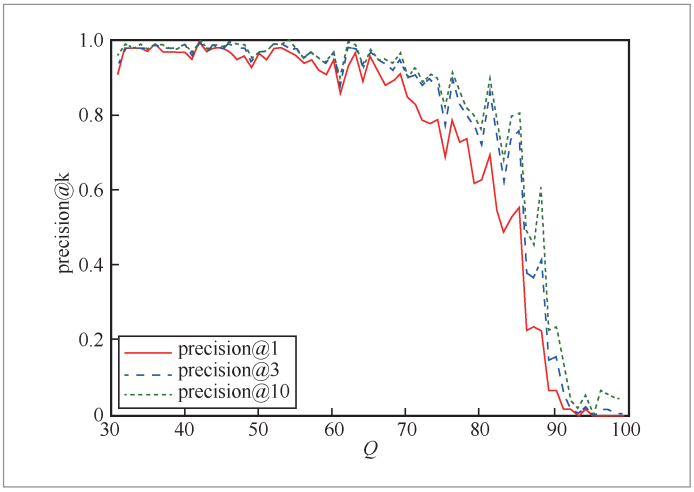

图 2 检索精确度与约束条件强度 Q 的关系

连通性越强。

● 近邻图的连通性对FGS算法检索精确度有决定性的影响。当近邻图的连通性较高时，检索精确度较高；反之，精确度较低，并趋向于0。因此改进检索路由中的连通性，可以提升检索结果的精确度。

3.2 FGS算法原理

FGS算法的基本思想是从初始查询向量集合ep出发，执行深度优先的贪心路由策略，不断地逼近查询向量 $\boldsymbol{x}_p$ 并记录遍历过的向量，从满足约束条件的已知向量中选择与 $\boldsymbol{x}_p$ 距离最近的前 k 个作为检索结果，其详细过程如算法1所示。

算法1：FGS.过滤贪心搜索算法.

输入：近邻图 G，导航向量集合ep，查询向量 $\boldsymbol{x}_q$，路由候选集合大小ef，结构化约束条件 c

输出：$\mathcal{L}$ 包含了 k 个 $\boldsymbol{x}_q$ 的近似最近邻；$\mathcal{V}$ 包含了所有访问过的向量信息

1：初始化：$\mathcal{L}\leftarrow\varnothing$，$\mathcal{V}\leftarrow\varnothing$ 。

2：FOR p IN ep:

3：$\mathcal{L}\leftarrow\mathcal{L}\cup\{p\}$

4：WHILE $\mathcal{L}\setminus\mathcal{V}\neq\varnothing$

5：令 $p^*\leftarrow\text{argmin}_{p\in\mathcal{L}\setminus\mathcal{V}}\text{dist}(\boldsymbol{x}_p,\boldsymbol{x}_q)$ # 路由候选集合中，未访问的距离 $\boldsymbol{x}_q$ 最近的向量

6：$\mathcal{V}\leftarrow\mathcal{V}\cup\{p^*\}$

7：$\mathcal{L}\leftarrow\mathcal{L}\cup\{p'\in N(G,p^*):c(p')=\text{Ture},p'\notin\mathcal{V}\}$

8：IF $|\mathcal{L}|$ > ef THEN

9：$\mathcal{L}\leftarrow\text{chooseNNearest}(\mathcal{L},\boldsymbol{x}_q,ef)$;

10：RETURN $[\text{chooseNNearest}(\mathcal{L},\boldsymbol{x}_q,k);\mathcal{V}]$

其中，$\text{dist}(\boldsymbol{x}_p,\boldsymbol{x}_q)$ 表示 $\boldsymbol{x}_p$、$\boldsymbol{x}_q$ 间距离度量的方法，例如欧氏距离、余弦距离等；$N(G,p)$ 表示在近邻图索引 G 中获取向量 p 的邻居列表；$\text{chooseNNearest}(\mathcal{L},\boldsymbol{x}_q,k)$ 表示从候选集合中获取距 $\boldsymbol{x}_q$ 最近的前 k 个向量。

在某一跳的搜索过程中，从当前跳的起始位置出发，将起始向量的邻居中符合结构化约束的向量记录到候选集合中。例如**图3（a）**中，假设当前跳为关键路径上的0号向量，那么本跳搜索过程中会将所有符合结构化约束的向量均标记为1（下一跳的序号），这些标记为1的向量会加入候选集合中。

遍历完关键路径上某个向量的所有邻居后，依据与 $\boldsymbol{x}_q$ 的距离压缩候选集合，选择距离 $\boldsymbol{x}_q$ 最近的且未访问过的向量作为下一跳起始位置。如**图3（a）**中关键路径上的1号向量即为下一跳的起始位置。

循环执行上述步骤，直到遍历完所有候选集合的向量。最后，将候选集合中距离 $\boldsymbol{x}_q$ 最近的前 k 个向量作为结果输出。

3.3 FGS算法问题分析

从上述算法流程中可以看出，候选集合在FGS算法中有两个重要用途：缓存可能的下一跳起始位置和缓存检索结果。在搜索过程中，由于FGS算法将两种功能均通过候选集合体现，候选集合在缓存下一跳候选位置时必须执行结构化约束条件，以满足缓存检索结果的目的，所以候选集合仅能增加满足 $\boldsymbol{x} \in P_C$ 的向量。因此，每一跳的候选向量均需要满足结构化约束。这会导致路由时近邻图的部分顶点不可达，索引的连通性下降。极端情况下，近邻图会形成多个不连通的子图，容易产生检索的局部性问题，从而降低检索结果的精确度。如**图3（a）**描述的场景中，由于候选集合仅包含向量 $\boldsymbol{x} \in P_C$，导致搜索时仅能通过近邻图的少部分边路由，容易陷入检索的局部区域，致使无法召回最近邻向量。**图3**中，近邻图 $m=16$，$\mathrm{ef}=5$，由30个向量构成，每个向量 $\boldsymbol{x}_i \in [0,1]^2$（$1 \leqslant i \leqslant 30$）。五角星表示 $\boldsymbol{x}_q$；红色点属于 P_C，满足结构化约束，蓝色方块表示不满足；黑色三角符号表示路由探索过的向量；黑色连线表示路由关键路径，点右上角的数字表示路由跳数，从0开始计数。

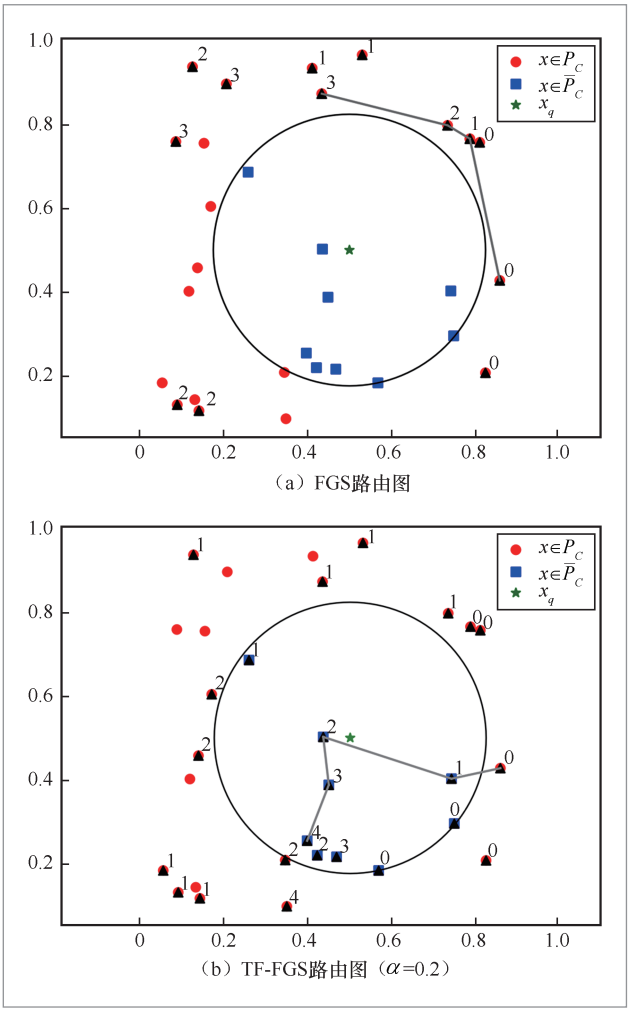

图 3 FGS 和 TF-FGS 路由路径选择策略对比（图中向量为随机生成的二维向量，横纵坐标为向量的值，取值范围均为 [0, 1]）

4 基于容忍因子的过滤贪心搜索算法

本文提出的TF-FGS算法在候选集合压缩时，允许点 $\boldsymbol{x} \in \overline{P_C}$ 参与检索路由，在不改变索引结构的条件下改善搜索过程中的

连通性，解决了FGS因结构化约束导致的连通性问题。如图3(b)所示，由于允许蓝色点（即 $x \in \overline{P_C}$ ）参与检索路由，能很大程度地利用近邻图的连通性，探索区域分布更丰富，避免了检索结构的局部性问题。

基于上述思路，TF-FGS算法在FGS算法的基础上将候选集合 $\mathcal{L}$ 拆分为路由候选集合 $\mathcal{L}'$（为了避免歧义，用 $\mathcal{L}'$ 表示TF-FGS的候选集合，包含了服从容忍因子 α 个数约束且满足 $x \in \cdot\overline{P_C}$ 的点）和结果候选集合 $\mathcal{R}$，在搜索路由过程中采取不同的策略进行迭代更新，详细过程见算法2和算法3。

算法2：TF-FGS. 基于容忍因子过滤贪心搜索算法.

输入：近邻图 G；导航点集合ep；查询向量 x_q，路由候选集合大小 ef；结构化约束条件 c；容忍因子 α

输出：$\mathcal{R}$ 包含 k 个近似最近邻点；$\mathcal{V}$ 包含所有访问过的节点

1: 初始化：$\mathcal{R} \leftarrow \varnothing$，$\mathcal{L}' \leftarrow \varnothing$，$\mathcal{V} \leftarrow \varnothing$。

2: FOR p IN ep:

3: $\mathcal{L}' \leftarrow \mathcal{L}' \cup \{p\}$

4: $\mathcal{R} \leftarrow \mathcal{R} \cup \{p\}$

5: WHILE $\mathcal{L}' \setminus \mathcal{V} \neq \varnothing$

6: 令 $p^* \leftarrow \text{argmin}_{p \in \mathcal{L} \setminus \mathcal{V}} \text{dist}(x_p, x_q)$

7: $\mathcal{V} \leftarrow \mathcal{V} \cup \{p^*\}$

8: $\mathcal{L}' \leftarrow \mathcal{L}' \cup \{p' \in N(G, p^*) : p' \notin \mathcal{V}\}$ # 增加路由候选点

9: $\mathcal{R} \leftarrow \mathcal{R} \cup \{p' \in N(G, p^*) : c(p') = \text{Ture}, p' \notin \mathcal{V}\}$

10: IF $|\mathcal{L}'| >$ ef THEN

11: $\mathcal{L}' \leftarrow \text{chooseNNearestWithToleren}(\mathcal{L}', x_q, \text{ef}, c, \alpha)$

12: IF $|\mathcal{R}| > k$ THEN

13: $\mathcal{R} \leftarrow \text{chooseNNearest}(\mathcal{R}, x_q, k)$

14: RETURN $[\mathcal{R}; \mathcal{V}]$

算法3：chooseNNearestWithTolerence. 改进的候选集合压缩算法

输入：候选集合 $\mathcal{L}'$；查询向量 x_q，路由候选集合大小ef；结构化约束条件 c；不满足结构化约束条件的容忍因子 α

输出：$\mathcal{S}$ 包含了 ef 个距离 x_q 的最近邻，且满足容忍因子 α 约束；

1: 初始化：$\mathcal{S} \leftarrow \varnothing$，$n\_\text{uncondition} = 0$ # 候选集合压缩后，不满足结构化约束的点的个数。

2: $\mathcal{L}' \leftarrow$ sort(dist($x_q, \forall x \in \mathcal{L}'$))

3: FOR p IN $\mathcal{L}'$:

4: IF len($\mathcal{S}$) = ef THEN

5: return $\mathcal{S}$

6: IF $c(p)$ THEN

7: $\mathcal{S} \leftarrow \mathcal{S} \cup \{p\}$

8: IF $c(p)$ = False AND $n\_\text{uncondition} <$ ef $* \alpha$ THEN

9: $\mathcal{S} \leftarrow \mathcal{S} \cup \{p\}$

10: $n\_\text{uncondition}$ += 1

4.1 候选集合迭代更新策略

TF-FGS算法在每次路由迭代中，路由候选集合 $\mathcal{L}'$ 会考虑所有新发现的点，然后使用基于容忍因子的 n 最近邻算法（chooseNNearestWithTolerence）进行压缩。与FGS算法不同的是，候选集合在压缩时会依据容忍因子 α 选择一定比例的满足 $x \in \cdot\overline{P_C}$ 的向量。

为了更好地控制允许点 $x \in \overline{P_C}$ 参与检索路由的比例，本文提出了容忍因子：

$$\alpha = 1 - \frac{|\{x \in \mathcal{L}' : C\}|}{\text{ef}} \qquad (6)$$

当 $\alpha = 0$ 时，压缩路由候选集合不会保留不满足结构化约束的点，等同于FGS算法；当 $0 < \alpha \leqslant 1$ 时，压缩路由候选集合会尽

可能以比例α保留不满足结构化约束的点。

在路由过程中，按照容忍因子α(0≤α≤1)来压缩候选集合$\mathcal{L}'$。候选集合$\mathcal{L}'$按照与$\boldsymbol{x}_q$的距离升序排序，压缩后的集合依次从排序集合中选择候选点，直到集齐ef个点；在选择过程中，如果$\boldsymbol{x}\in\overline{P_C}$且$n_\text{uncondition}>\text{ef}*\alpha$，则忽略该点。因此最多可以有$\alpha\times\text{ef}$个不满足结构化约束条件的点参与路由。

4.2 结果候选结合迭代更新策略

结果候选集合$\mathcal{R}$每次路由迭代增加满足$\boldsymbol{x}\in P_C$的点，压缩时直接选择距离查询点$\boldsymbol{x}_q$最近的前k个，迭代结束后作为结果返回。长度为k的结果集合$\mathcal{R}$记录了搜索的最终结果，结果集合在增加元素时执行结构化约束条件。

5 复杂度分析

5.1 FGS算法的复杂度分析

FGS算法的时间复杂度由搜索过程中检查的平均顶点个数、候选集合筛选、结构化约束计算3个部分组成。

假设图索引中，顶点的最大度为m，结构化约束评估的时间复杂度为$\mathrm{O}(e)$。每次查询平均会检查$\mathrm{O}(h\cdot o)$个向量，其中h表示平均每次查询的路由跳数，期望值为$\log N/m$；o（$o\leqslant m$）表示每跳需要检查的平均邻居个数，期望值为m。筛选候选集合的时间复杂度取决于不同的实现方法，使用最大堆进行筛选，单个向量筛选的时间复杂度为$\mathrm{O}(\text{ef}\cdot\log\text{ef})$。综上所述，由于在大规模查询场景下，$m$和ef远小于$N$，FGS的时间复杂度为$\mathrm{O}(e\cdot h\cdot o\cdot\text{ef}\cdot\log\text{ef})=\mathrm{O}(e\cdot m\cdot\text{ef}\cdot\log\text{ef}\cdot\log N/m)\approx\mathrm{O}(e\log N)$。特别地，当结构化约束评估的时间复杂度为$\mathrm{O}(e)\cong\mathrm{O}(1)$时，FGS的时间复杂度为$\mathrm{O}(\log N)$。

FGS的空间复杂度由候选集合和访问点集合两部分构成。其中，候选集合空间复杂度为$\mathrm{O}(\text{ef})=\mathrm{O}(1)$；访问点集合最差的空间复杂度为$\mathrm{O}(N)$，平均复杂度为$\mathrm{O}(h\cdot m)=\mathrm{O}(m\log N/m)\approx\mathrm{O}(\log N)$。综上所述，FGS的空间复杂度最差为$\mathrm{O}(N)$，平均复杂度为$\mathrm{O}(\log N)$。

5.2 TF-FGS算法的复杂度分析

TF-FGS算法在FGS算法的基础上改变了候选集合筛选的时间复杂度，增加了结果集合操作时间复杂度。

假设原始候选集合包含ef个元素，TF-FGS算法在实现候选集合筛选时，平均每跳增加元素o个，那么候选集合筛选的时间复杂度为$\mathrm{O}\left(\text{ef}\cdot\log(\text{ef}+o)\right)$。平均每跳增加元素$o$个，满足结构化约束条件的点一般少于或等于$o$个，那么结果集合操作的时间复杂度为$\mathrm{O}(\log o)$。综上所述，由于在大规模查询场景下，$m$和ef远小于$N$，TF-FGS算法的时间复杂度为$\mathrm{O}\left(e\cdot h\cdot o\cdot\left(\text{ef}\cdot\log(\text{ef}+o)+\log o\right)\right)\approx\mathrm{O}(e\log N)$。特别地，当结构化约束评估的时间复杂度为$\mathrm{O}(e)\cong\mathrm{O}(1)$时，FGS的时间复杂度为$\mathrm{O}(\log N)$。因此TF-FGS算法引入容忍因子后，时间复杂度与FGS算法在同一级别。

TF-FGS算法在FGS算法的基础上增加了结果集合存储，该部分的空间复杂度为$\mathrm{O}(k)$，为常量开销。因此，TF-FGS算法的平均空间复杂度仍为$\mathrm{O}(\log N)$，最差为$\mathrm{O}(N)$。

6 实验与分析

本文对比了FGS算法和TF-FGS算法

两种搜索算法混合查询时在不同索引、不同数据集上的性能表现，以验证本文提出的TF-FGS算法的有效性。本文使用了3个公开数据集（GloVe.6B[26]、SIFT1M[29]和Deep1B[30]），并评估了本算法在数据上的准确率和查询时间。通过实验，旨在找到一种支持混合查询的近邻图搜索算法，以满足实际的应用需求。

6.1 实验设置

本文选用了多样化的ANNS基准数据集进行实验，包括GloVe.6B[26]、SIFT1M[29]和Deep1B[30]，数据集的详细说明见**表1**。这些数据集在数据维度、向量来源类型等方面具有显著差异，从而使实验结果具有较强的代表性和普适性。

GloVe.6B数据集来源于自然语言处理领域，它是一种基于全局词向量表示的方法，用于捕捉词汇之间的语义关系。SIFT1M数据集则是一个常用的计算机视觉数据集，包含了大量局部特征描述符，有助于理解图像内容。Deep1B数据集来自深度学习领域，包含了海量的特征向量，用于描述复杂的数据结构。多元化的数据来源有助于确保实验结果具有广泛的普适性，从而为计算机信息检索领域提供有价值的参考。

由于上述数据集中无结构化约束属性，本文遵循前人的研究工作经验[8,14]，对数据集模拟改造，模拟混合查询场景数据。具体地，利用随机数合成约束属性，随机数可以使用式（7）计算：

$$r = \lfloor U \cdot N \rfloor \tag{7}$$

表1 数据集说明

名字	发布时间	类型	规模	维数	约束属性
Deep1B	2016年	图片	1 B	96	合成
GloVe.6B	2014年	文本	6 B	50	合成
SIFT1M	2010年	图片	1 MB	128	合成

其中，U 是一个取值在 $[0,1]$ 区间上的均匀分布随机变量；N 表示数据集的大小；$\lfloor \cdot \rfloor$ 表示向下取整函数。

本文中与检索路由相关的参数主要有候选集合大小ef、容忍因子 α。其中ef的取值为4、8、16、32、64；a 取值为0、3、6、10。

本文使用了两个主要的评估指标，分别是检索精度和检索效率。

对于检索精度，本文使用 precision@k 作为评估指标。在每个数据集上，对于不同的实验参数，选择随机的点作为查询向量。每次将算法返回的前 k 个结果作为检索结果，其中 k 的值为1、3、10。对于检索效率，本文使用检索时间作为评估指标。

6.2 TF-FGS算法与FGS算法效果对比

本文选择实验参数 $\alpha = 0.3$、$\text{ef} = 64$，使用precision@10表示检索效率，在3个公开数据集上评估结构化约束强度 Q 取值为30%、60%、90%时TF-FGS算法与FGS算法混合查询的效果，结果见**表2**。

从**表2**可观察出，在所有数据集中，使用TF-FGS算法比FGS算法在相同的结构化约束强度下 precision@10 均有明显提高。这表明本文提出的基于容忍因子的过滤贪心算法可以有效地克服结构化约束条件对检索精度的负面影响。

在DEEP1B数据集中，当 $Q = 30\%$ 时，使用TF-FGS比FGS精度提升了10%左右；当 $Q = 90\%$ 时，精度提升了53.3%。这说明在数据集的结构化约束比较弱时，使用TF-FGS算法比FGS算法提高的精度较小；而在数据集的结构化约束比较强时，使用TF-FGS算法比FGS算法提高的精度就会更加明显。在GloVe.6B和SIFT1M数据集中也存在类似的规律。

这表明在不同的数据集中，本文提出

的算法都可以有效地提高精度，并且随着结构化约束强度增加，相对FGS算法对检索精度的提升越大。从**表2**可知，TF-FGS算法检索结果的 precision@10 相比FGS算法平均提升了30%。

表 2 TF-FGS 算法与 FGS 算法搜索效果对比

Q	查询方法	DEEP1B	GloVe.6B	SIFT1M	精度提升
30%	FGS	0.63	0.53	0.8	10%
	TF-FGS	0.69	0.57	1.0	
60%	FGS	0.52	0.34	0.20	24%
	TF-FGS	0.64	0.54	0.60	
90%	FGS	0.02	0.02	0.00	53.3%
	TF-FGS	0.73	0.41	0.5	

6.3 容忍因子对混合查询的影响

本实验固定了检索类型和数据类型，采用GloVe.6B数据集，使用HNSW的默认参数构建索引，并随机选择查询向量进行实验。在保持索引结构不变的条件下讨论不同实验超参数对TF-FGS检索精确度的影响。在GloVe.6B数据集上检索参数 ef 、α 和 Q 对 precision@k 的影响如**图4**所示。

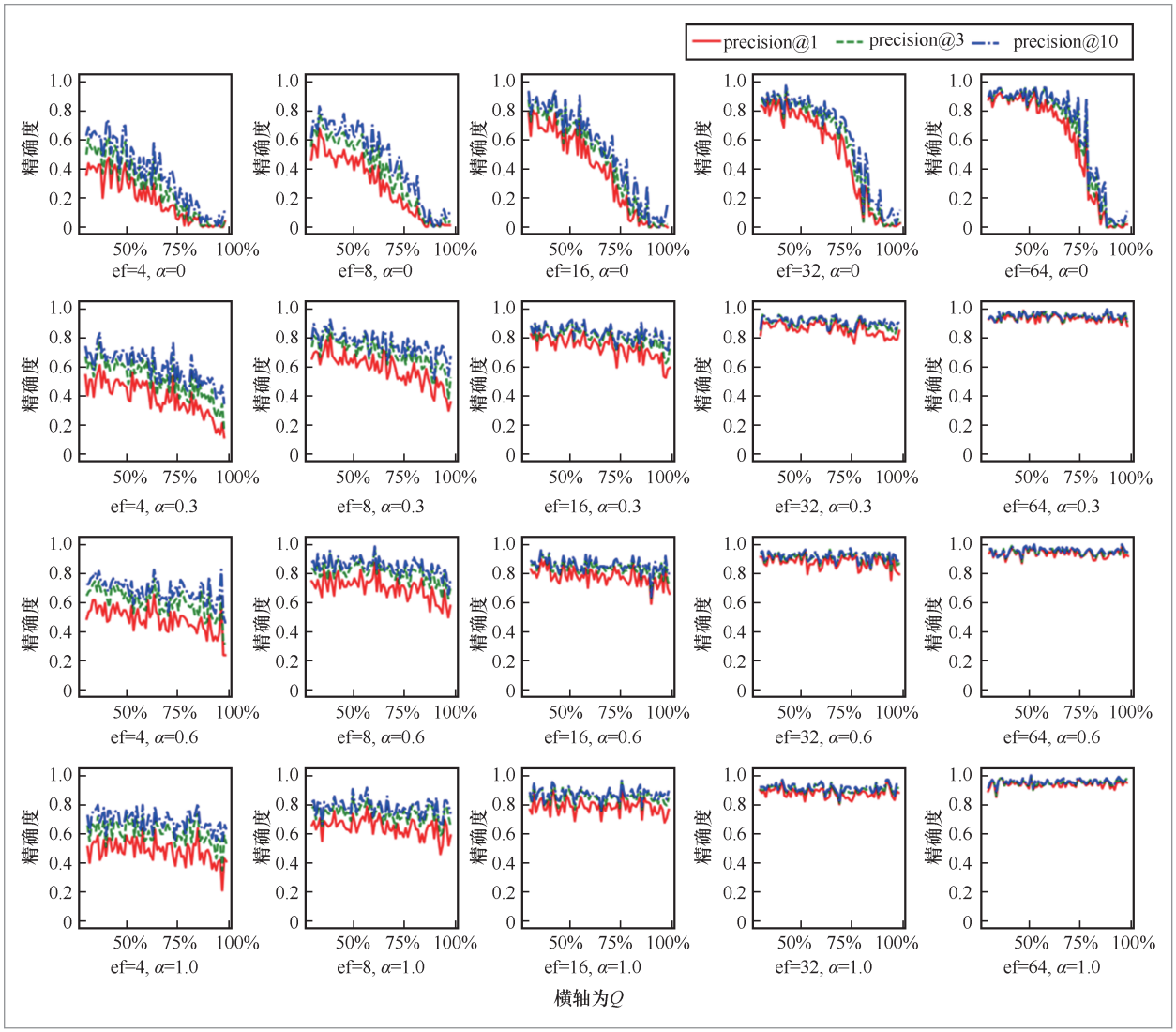

图 4 GloVe.6B 上近邻图的连通性对检索精确度的影响

同等参数条件下近邻图的连通性对检索效率的影响如**图5**所示。

观察**图4**、**图5**，可以发现以下结论。

- 参数 ef 不变，容忍因子 $\alpha=0$ 时（FGS算法），在满足过滤条件的点占比逐渐下降时，检索精确度会随之下降，特别是在过滤掉85%以上的点时，精确度趋于0。但此时检索效率会极大提升，这是由于近邻图不连通，导致提前结束查询。
- 参数 ef 不变，容忍因子 $\alpha>0$ 时，允许不满足结构化约束的点参与路由，有效地缓解了检索精确度随满足过滤条件的向量占比下降而逐渐降低的问题。而此时检索耗时相对稳定。
- 容忍因子 α 不变时，候选集合大小 ef 决定了检索精确度的上限，且随着 ef 的增加，检索精确度的提升越来越少。
- 在结构化约束强度 Q 较大时，引入容忍因子 α 对查询精确度的提升越明显。

综上所述，在不改变索引结构的条件下，检索参数 ef 不变时，适当的容忍因子 α 可以在保持检索效率不变的同时，显著

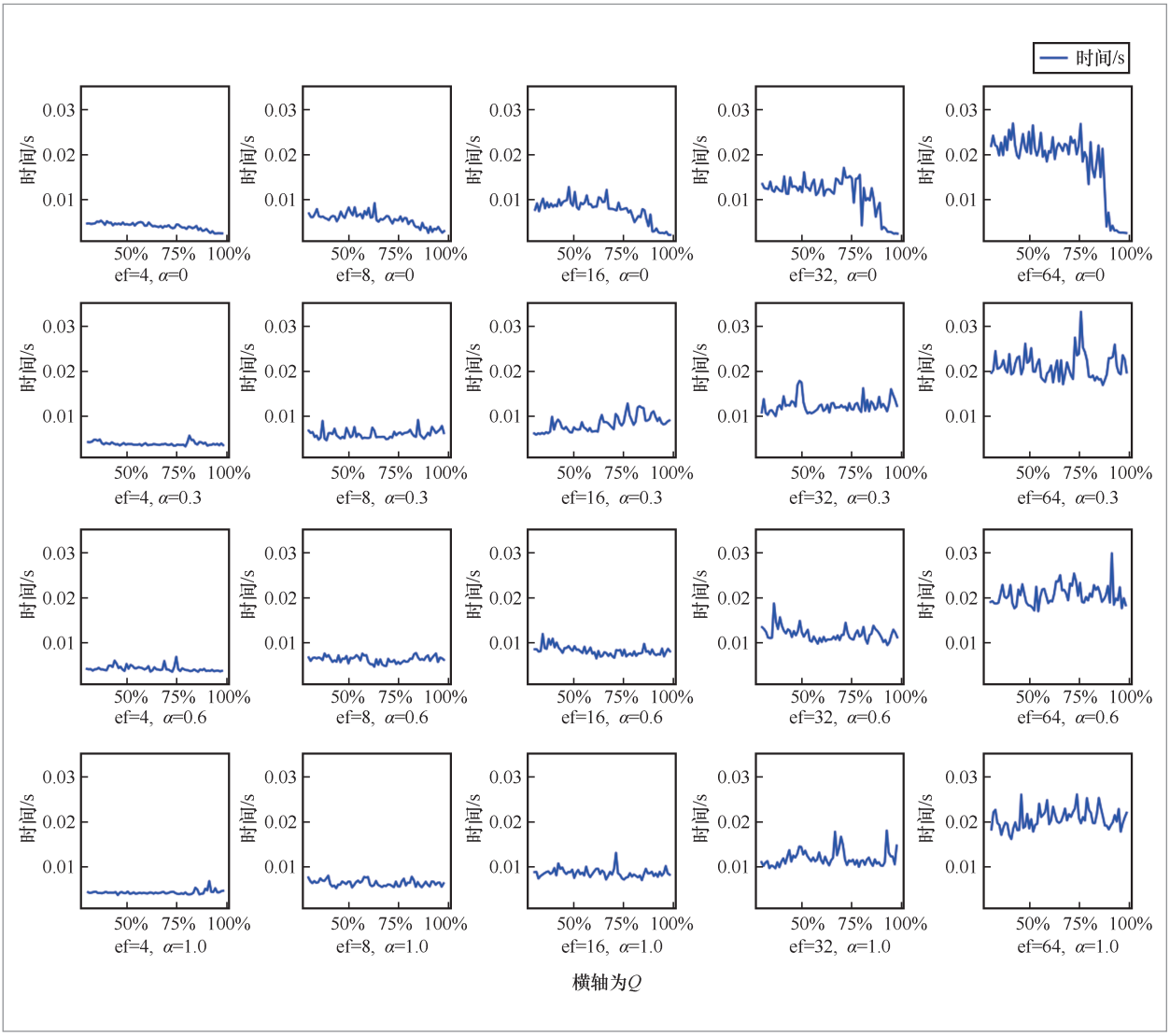

图 5 GloVe.6B 上近邻图的连通性对检索效率的影响

增强检索的精确度，且在结构化约束强度 Q 较大时尤为显著。

6.4 TF-FGS算法在不同索引上的有效性分析

本实验验证了TF-FGS算法在多种不同的近邻图索引上相对于FGS算法能够获得更好的检索性能，两者的效果对比如图6所示。

本实验使用SIFT1M数据集，并使用HNSW、NSG、SSG 3种索引的默认参数建图，设定容忍因子为0.3。在数据集和索引结构不变的条件下，比较了不同结构约束强度 Q（取值范围为31%~99%）下FGS算法和TF-FGS算法的检索效果。

从结果可以观察到，虽然在HNSW、NSG、SSG 3种索引上应用同一混合查询方法时检索精度有不同程度的差异，但是索引一定的条件下，当结构约束 Q 过强时，应用FGS算法查询检索精度会快速下降并趋近于0；而应用TF-FGS算法查询，无论索引结构如何，检索精度均能维持在较高水平。这表明在不同的近邻图索引上，TF-FGS算法相对于FGS算法能够获得更优的检索精确度。

6.5 TF-FGS算法在不同数据集的有效性分析

本实验使用HNSW在DEEP1B、GloVe.6B和SIFT1M数据集上分别构建近邻图索引，分析TF-FGS算法在不同数据集、不同的超参数取值时混合查询的有效性。其中，超参数包括候选集合容量ef、容忍因子α、结构化约束强度Q。多数据集上近邻图的连通性对检索精度和检索效率的影响分别如图7和图8所示。

从图7、图8可以看出，无论数据集和超参数设置如何，TF-FGS算法在保持近似

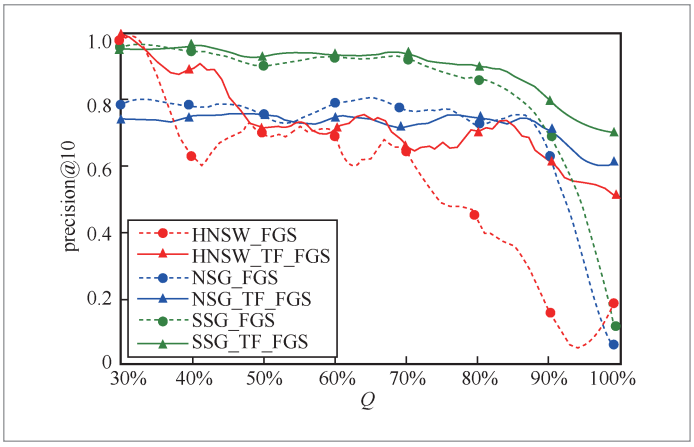

图 6 不同索引中 TF-FGS 算法和 FGS 算法的效果对比

相同的检索效率的前提下均能够获得明显更高的精确率。这证明了TF-FGS算法相较于FGS算法在不损耗效率的条件下可以提供更准确和更可靠的近似邻域搜索结果。

如图8所示，当参数 ef 一定时，容忍因子 α 不宜过大。虽然过大的 α 不会降低检索精确度，但是会导致检索效率不稳定。因为过大的 α 会使更多不满足结构化约束的点参与路由，当 x_q 最近邻附近有更多符合条件的点时，其会有较高的检索效率；反之，则会搜索更多的无效分枝，进行更多的距离计算和对比，从而造成检索效率降低。

综上所述，实验部分证明了TF-FGS算法在多种近邻图索引和多种数据集下均有较好的表现，具有一定的普适性，其可以在不影响检索效率的前提下解决路由时的连通性问题，极大地降低了结构化约束强弱对检索精确度的影响。

7 总结与展望

经过实验验证，本文改进的过滤贪心搜索算法能够在不影响检索效率的前提下提升近似最近邻检索的精确度，具有一定

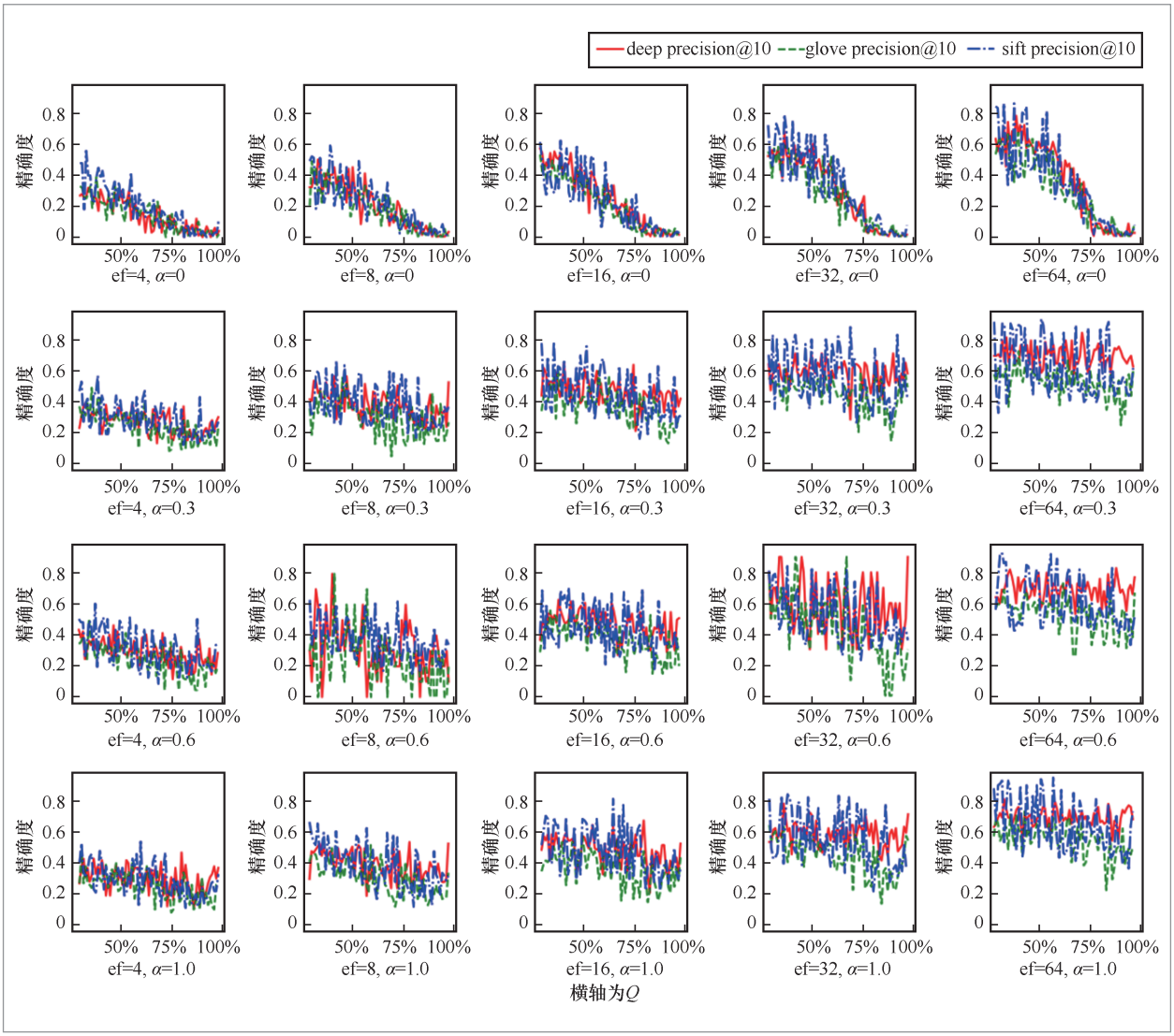

图 7 多数据集上近邻图的连通性对检索精确度的影响

的有效性和普适性。然而，这种方法仅仅消除了结构化约束条件对检索精确度的影响，如何在此基础上利用结构化约束条件缩小检索范围、提升检索效率仍然是一个值得探究的问题。未来可能的研究方向包括利用树结构索引来加速搜索过程，提高检索效率和精度；也可以考虑将本方法与其他近似最近邻搜索算法相结合，探索出更加高效、准确的检索方法。总之，本文提出的方法为近似最近邻检索领域的混合查询研究提供了新思路，未来还有很多有意义的工作可以展开。

参考文献:

[1] YANG F, HINAMI R, MATSUI Y, et al. Efficient image retrieval via decoupling diffusion into online and offline processing[J]. Proceedings of the AAAI Conference on Artificial Intelligence,

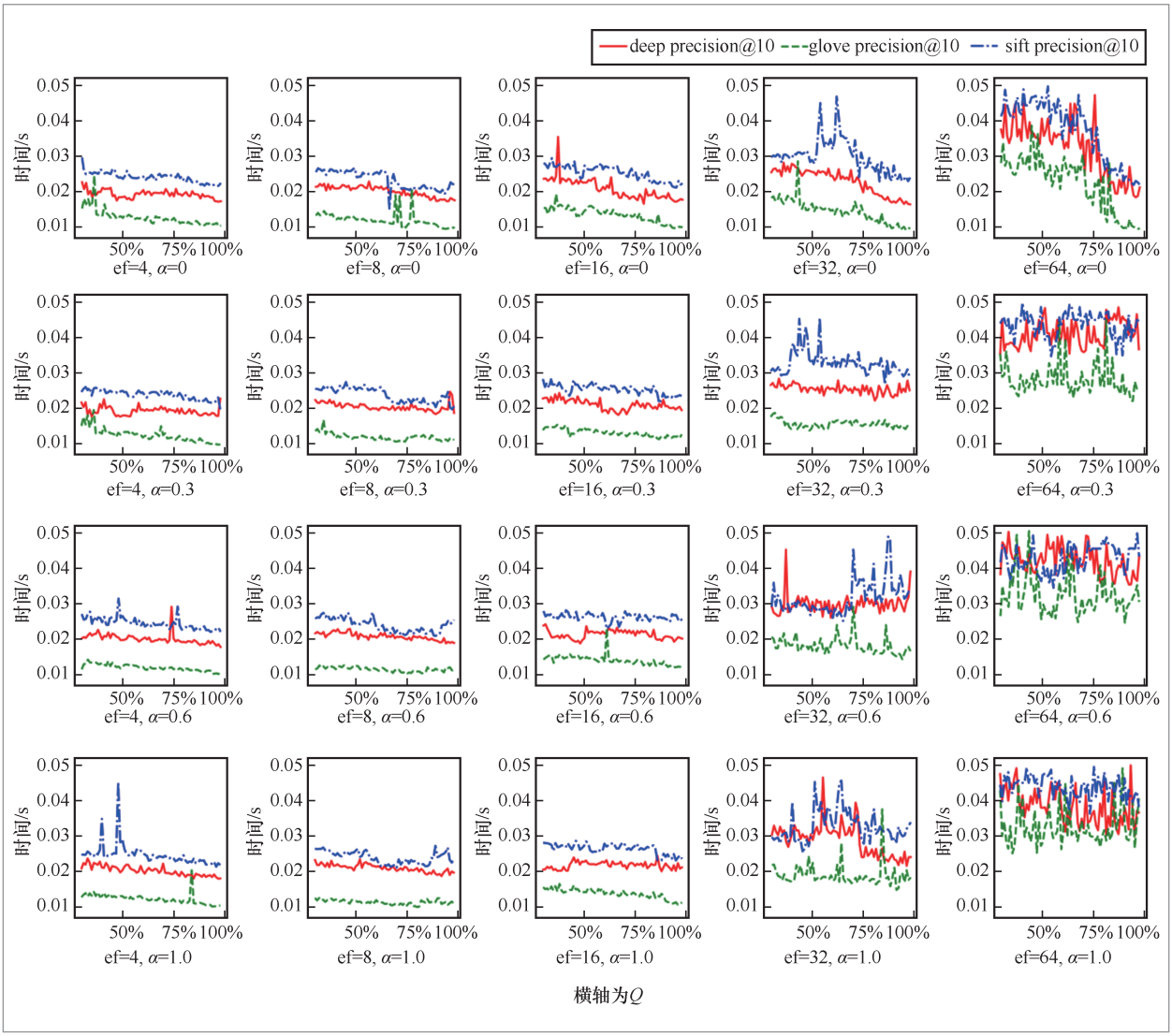

图 8 多数据集上近邻图的连通性对检索效率的影响

2019, 33(1): 9087–9094.

[2] PHILBIN J, CHUM O, ISARD M, et al. Object retrieval with large vocabularies and fast spatial matching[C]//Proceedings of 2007 IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE Press, 2007: 1–8.

[3] SHANKAR D, NARUMANCHI S, ANANYA H A, et al. Deep learning based large scale visual recommendation and search for E–commerce[EB]. arXiv preprint, 2017, arXiv: 1703.02344.

[4] CHEN R H, LIU B, ZHU H, et al. Approximate nearest neighbor search under neural similarity metric for large–scale recommendation[C]//Proceedings of the 31st ACM International Conference on Information & Knowledge Management. New York: ACM, 2022: 3013–3022.

[5] BERLIN K, KOREN S, CHIN C S, et al. Assembling large genomes with single–molecule sequencing and locality–sensitive hashing[J]. Nature Biotechnology, 2015, 33(6): 623–630.

[6] TOPSAKAL O, AKINCI T C. Creating large language model applications utilizing

LangChain: a primer on developing LLM apps fast[J]. International Conference on Applied Engineering and Natural Sciences, 2023, 1(1): 1050–1056.

[7] AUMÜLLER M, BERNHARDSSON E, FAITHFULL A. ANN-Benchmarks: a benchmarking tool for approximate nearest neighbor algorithms[J]. Information Systems, 2020, 87: 101374.

[8] GOLLAPUDI S, KARIA N, SIVASHANKAR V, et al. Filtered-DiskANN: graph algorithms for approximate nearest neighbor search with filters[C]//Proceedings of the ACM Web Conference 2023. New York: ACM, 2023: 3406–3416.

[9] WANG J G, YI X M, GUO R T, et al. Milvus: a purpose-built vector data management system[C]//Proceedings of the 2021 International Conference on Management of Data. New York: ACM, 2021: 2614–2627.

[10] VAN LUIJT B, VERHAGEN M. Bringing semantic knowledge graph technology to your data[J]. IEEE Software, 2020, 37(2): 89–94.

[11] LI J, LIU H F, GUI C H, et al. The design and implementation of a real time visual search system on JD E-commerce platform[C]//Proceedings of the 19th International Middleware Conference Industry. New York: ACM, 2018: 9–16.

[12] JOHNSON J, DOUZE M, JEGOU H. Billion-scale similarity search with GPUs[J]. IEEE Transactions on Big Data, 2021, 7(3): 535–547.

[13] PARK Y, PARK S, LEE S-G, et al. Greedy filtering: a scalable algorithm for k-nearest neighbor graph construction[C]//Proceedings of the Database Systems for Advanced Applications: 19th International Conference. Berlin: Springer, 2014: 327–41.

[14] WANG M, LYU L, XU X, et al. Navigable proximity graph-driven native hybrid queries with structured and unstructured constraints[EB]. arXiv preprint, 2022, arXiv: 2203.13601.

[15] 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.

[16] STREHL A, GHOSH J. Cluster ensembles: a knowledge reuse framework for combining multiple partitions[J]. Journal of Machine Learning Research, 2003, 3: 583–617.

[17] AGGARWAL C C. Data classification: algorithms and applications[M]. [S.l.]: CRC Press, 2014: 37–64.

[18] XU X L, LI C, WANG Y X, et al. Multiattribute approximate nearest neighbor search based on navigable small world graph[J]. Concurrency and Computation: Practice and Experience, 2020, 32(24): e5970.

[19] BECKMANN N, KRIEGEL H P, SCHNEIDER R, et al. The R*-tree: an efficient and robust access method for points and rectangles[C]//Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data. New York: ACM, 1990: 322–331.

[20] MUNAGA H, JARUGUMALLI V. Performance evaluation: ball-tree and KD-tree in the context of MST[C]//Proceedings of International Joint Conference on Advances in Signal Processing and Information Technology. Heidelberg: Springer, 2012: 225–228.

[21] GREENSPAN M, YURICK M. Approximate k-d tree search for efficient ICP[C]//Proceedings of 4th International Conference on 3-D Digital Imaging and Modeling. Piscataway: IEEE Press, 2003: 442–448.

[22] DOLATSHAH M, HADIAN, MINAEI-BIDGOLI B. Ball*-tree: efficient spatial indexing for constrained nearest-neighbor search in metric spaces[EB]. arXiv preprint, 2015, arXiv: 1511.00628.

[23] MALKOV Y A, YASHUNIN D A. Efficient

and robust approximate nearest neighbor search using hierarchical navigable small world graphs[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020, 42(4): 824-836.

[24] FU C, XIANG C, WANG C X, et al. Fast approximate nearest neighbor search with the navigating spreading-out graph[J]. Proceedings of the VLDB Endowment, 2019, 12(5): 461-474.

[25] FU C, WANG C X, CAI D. High dimensional similarity search with satellite system graph: efficiency, scalability, and unindexed query compatibility[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022, 44(8): 4139-4150.

[26] PENNINGTON J, SOCHER R, MANNING C. Glove: global vectors for word representation[C]//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. Stroudsburg: Association for Computational Linguistics, 2014: 1532-1543.

[27] ERDŐS P, RÉNYI A. On random graphs[J]. Publications Mathematica Debrecen, 2022, 6(3/4): 290-297.

[28] WATTS D J, STROGATZ S H. Collective dynamics of "small-world" networks[J]. Nature, 1998, 393(6684): 440-442.

[29] JÉGOU H, DOUZE M, SCHMID C. Improving bag-of-features for large scale image search[J]. International Journal of Computer Vision, 2010, 87(3): 316-336.

[30] YANDEX A B, LEMPITSKY V. Efficient indexing of billion-scale datasets of deep descriptors[C]//Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE Press, 2016: 2055-2063.

作者简介

贺广福(1992-),男,中国科学院大学博士生,中国科学院计算技术研究所工程师、中级软件设计师,主要研究方向为大数据、信息检索、自然语言处理。

薛源海(1987-),男,博士,中国科学院计算技术研究所高级工程师,主要研究方向为信息检索。

陈翠婷(1991-),女,中国科学院大学博士生,中国科学院计算技术研究所工程师,主要研究方向为人工智能、信息检索。

俞晓明（1977– ），男，博士，中国科学院计算技术研究所高级工程师，主要研究方向为信息检索、自然语言处理。

刘欣然（1971– ），男，博士，北京邮电大学研究员，主要研究方向为网络空间安全、大数据应用。

程学旗（1971– ），男，博士，中国科学院计算技术研究所研究员、副所长，主要研究方向为网络科学与社会计算、网络大数据。

收稿日期： 2023-08-30
基金项目： 国家自然科学基金项目（No.U21B2046）
Foundation Item: The National Natural Science Foundation of China (No.U21B2046)